Metadata Provenance in Europeana and the Semantic Web

M A S T E R T H E S I S

submitted in partial fulfillment of the requirements for the degree of

Master of Arts
in Library and Information Science

at
Faculty of Arts I
Humboldt-Universität zu Berlin

by
Dipl.-Inf., Dipl.-Wirt.-Inf. Kai Eckert

President of the Humboldt-Universität zu Berlin:
Prof. Dr. Jan-Hendrik Olbertz

Dean of the Faculty of Arts I:
Prof. Michael Seadle, PhD

Referees:
1. Prof. Dr. Stefan Gradmann
2. Prof. Dr. Christian Bizer

Submitted: May 25, 2012
Oral exam: Sep 17, 2012

Contents

Acknowledgements

I would like to thank all organizers and participants of the W3C Provenance Incubator Group and the W3C Provenance Working Group, who invited me as an expert. I learned much more than I had to provide.

I thank the Dublin Core Metadata Initiative who embraced me and welcomed my ideas since I first showed up talking about metametadata. Special thanks to the DCMI Architecture Forum, namely Corey Harper and Tom Baker, who motivated me to invest my rare time into strange metadata models and at the same time showed me the way to make my efforts productive and worthwhile.

Very special thanks, of course, to Michael Panzer and Daniel Garijo, my colleagues in the DCMI Metadata Provenance Task Group. Everything substantial inside this thesis was probably figured out during our weekly phone calls. They have always been a pleasure, as well as insightful and inspiring.

1 Introduction

At the toolbar (menu, whatever) associated with a document
there is a button marked "Oh, yeah?".
You press it when you lose that feeling of trust.
It says to the Web, "so how do I know I can trust this information?".
The software then goes directly or indirectly back to
metainformation about the document, which suggests a number of reasons.

Tim Berners-Lee[1]

According to Merriam-Webster,[2] *provenance* is the "origin, source" or the "history of ownership of a valued object or work of art or literature." This definition illustrates in which context the term provenance has been and still is used – to describe the provenance of artworks, with the goal to prove legitimate ownership and originality of the works.

Provenance, however, is not limited to artworks and their ownership; there are many reasons why one would like to know the origin of arbitrary resources. For instance, in the automotive industry the information is needed which cars are affected by a fault in the production process, if such a fault is detected. In the web of data, the origin of a certain information is crucial to judge its correctness and suitability for a potential reuse in a different application. Provenance information is a requirement for trust. We do not only need the provenance of valued objects; in particular data objects can only become valued if provenance information is available.

For our purpose the definition of Merriam-Webster is therefore not suitable. In this thesis, we use the definition of the W3C Provenance Incubator Group (2010):

Definition 1.1 (Provenance) *Provenance of a resource is a record that describes entities and processes involved in producing and delivering or otherwise influencing that*

[1] http://www.w3.org/DesignIssues/UI.html
[2] http://www.merriam-webster.com/dictionary/provenance

resource. Provenance provides a critical foundation for assessing authenticity, enabling trust, and allowing reproducibility. Provenance assertions are a form of contextual metadata and can themselves become important records with their own provenance.

Remarkable is the inclusion of the processes and their description that lead to the creation (or modification) of a resource. This distinguishes provenance information adhering to this definition from mere ownership or creator statements that are often regarded as provenance information. We will come back to this distinction later in this thesis. Equally important is the characterization of provenance information as metadata and the notion that such metadata can have its own provenance. This is the topic of this thesis: *metadata provenance*.

Metadata – descriptive data about resources, including, but not limited to provenance data – is widely used in libraries to describe and organize the resources that have to be made available through a library. Bibliographic metadata is often created collaboratively and reused in other libraries, but also outside the libraries. Additionally, metadata from diverse sources is used, for example from publishers or from specialized metadata providers. Various standards and formats exist to represent metadata, like MARC 21[3] or Dublin Core,[4] to name two prominent representatives that show the bandwidth and different focusings: MARC 21 as the current de-facto standard for the exchange of bibliographic metadata in detail, Dublin Core as a very simple and general way to provide metadata with the focus on cross-domain interoperability of metadata applications. This leads to a rich, but complex and heterogenous metadata environment, where various different descriptions of different resources exist, with inevitable doublets – or redundancies, depending on the point of view.

The World Wide Web provides an infrastructure to exchange metadata between applications and users. The *Resource Description Framework* (RDF) (RDF Core Working Group, 2004a) provides functionality to prevent ambiguities between different datasets and is widely used to represent data on the Web. Additionally, RDF provides various means to relate the data on different levels and by means of these relationships foster the proper interpretation of heterogenous data in applications. The data becomes part of the Web and forms a Web of Data: the *Semantic Web*.

The name "Resource Description Framework" indicates that RDF data is generally metadata, albeit in the broadest possible sense, as a resource in RDF can be – almost –

[3] http://www.loc.gov/marc/bibliographic/
[4] http://dublincore.org/documents/dcmi-terms/

everything. A notable exception is RDF data itself, currently it is at least not clear how to speak about RDF data within RDF. This, however, is needed to represent metadata about metadata, including metadata provenance. Nevertheless, there are ways and best-practices how to represent metadata provenance.

The use case for metadata provenance employed in this thesis is the *Europeana Data Model* (EDM) (Europeana, 2011, 2012; Doerr et al., 2010). Europeana, the European digital library, collects and exposes metadata from European cultural heritage institutions, particularly libraries, archives and museums. The EDM is formulated in RDF and thus directly brings the Europeana data into the Semantic Web. One problem that the EDM has to deal with is that for one resource, typically more than one description from more than one institution is available. The distinction of these sources and the correct attribution of an institution for a description is a strong requirement for the EDM. We will show how this is currently accomplished in EDM and what alternative ways for the provenance representation would be possible.

1.1 Research Questions and Contributions

This thesis builds on work of the World Wide Web Consortium (W3C) and the Dublin Core Metadata Initiative (DCMI). We give an overview on various ways to handle metadata provenance in the Semantic Web. The necessary foundations are provided in the next chapter. In Chapter 3, we report the current state of the W3C Provenance Working Group that develops an RDF based provenance data model. Furthermore, we introduce Dublin Core as a simple provenance vocabulary and relate it to the Provenance WG model. The core of this thesis is about metadata provenance (Chapter 4). We describe current best-practice solutions and provide an overview about the current developments in the DCMI Metadata Provenance Task Group. Furthermore we take into account the current developments regarding graph identification in the W3C RDF Working Group. Based on these foundations, we develop a proposal for the Europeana Data Model in Chapter 5. The thesis is concluded with a discussion of our approach and some thoughts regarding further implementation steps.

1.1.1 Research questions

Particularly, we aim at answering the following research questions:

1. How do metadata models like Dublin Core relate to more complex provenance models?
2. Is it possible to provide a mapping between them?
3. What are the general problems of metadata provenance?
4. How does a graph based identification of metadata records affect the representation of metadata provenance?
5. Would such an approach be advantageous for the EDM?
6. Would the use of a complex provenance model be advantageous for the EDM?

1.1.2 Contributions

The contributions of this thesis are the following:

1. Review of the W3C Provenance WG.
2. Review of the DCMI Metadata Provence TG.
3. Review of the W3C RDF WG.
4. Provision of a proposal for an extended abstract metadata model suitable for provenance representation.
5. Provision of a mapping strategy between Dublin Core and the upcoming W3C provenance model.
6. Reformulation of the EDM using graph based identification of metadata records.
7. Critical review and guidance on further steps, if the EDM is to be revised.

The proposal for an extended abstract metadata model suitable for provenance representation was developed by Kai Eckert, Daniel Garijo, and Michael Panzer in the DCMI Metadata Provenance TG and is published in (Eckert, Garijo, & Panzer, 2011). The description in this thesis is revised and takes further developments and ideas from the DCAM revision into account that is currently performed in the DCMI Architecture Forum. The mapping strategy is currently developed by Kai Eckert, Daniel Garijo, Simon

Miles, and Michael Panzer in a joint effort of the DCMI Metadata Provenance TG and the W3C Provenance WG. The final mapping will be part of the resulting deliverables of the W3C Provenance WG.

1.2 Scope and Limitations

This thesis is intended as a summary of current developments regarding the representation of metadata provenance in the Semantic Web. It is clearly limited due to the fact that none of the relevant working groups (W3C Provenance Working Group, W3C RDF Working Group, DCMI Metadata Provenance Task Group, DCMI Architecture Forum) have finished their work yet. In this regard, all results in this thesis can only be preliminary. The developments of these groups are, however, more or less stable and it can be assumed that most of the concepts that we use will be available in one way or the other in the near future.

The author of this thesis is involved in all but the RDF Working Group and the thesis reflects the most current results of these groups. We do not describe everything in all details but aim at providing a bigger picture how the working group results can be related and how this could lead to a stable, standardized model for metadata provenance.

This thesis only deals with the technical representation of metadata provenance. All other questions about the proper representation of provenance, in particular political and legal questions, are not subject of this thesis. Therefore, they are also not considered regarding the reformulation of the Europeana Data Model.

2 Metadata and the Semantic Web

Data is a precious thing and will last longer than the systems themselves.

Tim Berners-Lee[1]

Metadata is often defined as data about data. This is misleading, as for instance in libraries, all data about books in the library catalog is called metadata. While books could be seen as a form of data, latest with descriptions of other resources like the physical holdings of a museum, the definition does not fit any more. The greek prefix *meta* means *beyond*, *about*, or *among*. It is more useful to see metadata as *about-data*, i.e. data about something, or *beyond-data*, i.e., data that is beyond something, which implies that the data is somehow decoupled from the thing and resides on a different level, the meta level.

We therefore define metadata as follows:

Definition 2.1 (Metadata) *Metadata is structured data that is used to describe the properties of a resource.*

In the remainder of this chapter, we briefly introduce the foundations of the Semantic Web – the ecosystem for linked metadata embedded in the World Wide Web (WWW).

2.1 Resource Description Framework

The Resource Description Framework (RDF) provides a machine-interpretable language to represent information about resources in the WWW (RDF Core Working Group, 2004a, 2004b, 2004c, 2004d, 2004e). It is accompanied by the RDF Schema language

[1] (Runciman, 2006)

(RDFS). In RDF, everything is a resource. In particular, the class of resources (`rdfs:Resource`) is defined as follows in RDFS:

Definition 2.2 (Resource, `rdfs:Resource`) *All things described by RDF are called resources, and are instances of the class `rdfs:Resource`. This is the class of everything. All other classes are subclasses of this class.*

Information about resources is expressed in *statements* about the resource:

Definition 2.3 (Statement) *A statement is a triple, consisting of a subject, a predicate, and an object. A statement generally describes one property (given by the predicate) of one identifiable resource (given by the subject) by assigning one value (given by the object). Subject and object can be left unspecified, which simply indicates the* existence *of something, without using, or saying anything about, the name of that thing. The object can be another identifiable resource or a literal, i.e., for instance a character string, a number or a date.*

Note, that according to this definition, the predicate can *not* be left unspecified. This is a restriction in RDF that we incorporated in this definition for consistency. This means, that it is not possible in RDF to make a statement like "There exists a predicate that relates 324 meters to the Eiffel tower." Valid statements are "The Eiffel tower has a height of 324 meters." or "There is something that has a height of 324 meters."

The second restriction for statements in RDF is that statements can *not* be made about literals. This means that it is not possible to make a statement like "324 meters is the height of the Eiffel tower." This is generally only a small restriction, as the inverse statement from above is valid and semantically equivalent. However, this restriction means that it is not possible in RDF to relate two literals, like "324 meters are more than 200 meters."

To make statements machine-interpretable, they have to be further formalized. RDF distinguishes three types of resources that can be used to create statements: *RDF URI references*, *blank nodes* and *literals*.

RDF URI Reference: An RDF URI reference is a Uniform Resource Identifier according to RFC 2396 (Berners-Lee, Fielding, & Masinter, 1998), that identifies a resource. URIs are globally unique and every URI identifies one and only one resource. A resource, however, can be identified by more than one URI. URIs can

be classified as a locater, a name, or both. Uniform Resource Locators (URL) identify resources via a representation of their primary access mechanism, e.g., `http://example.org/resource1`. Uniform Resource Names (URN) provide a globally unique name for a resource independent of a location or access mechanism, e.g., `urn:example:resource1`. A URI always has the form `<scheme>:<scheme-specific-part>`, where `<scheme>` determines the type of the URI. The form of the scheme-specific part depends on the scheme, but always contains a mechanism to ensure the global uniqueness of the URI. For http-URLs, this is accomplished by means of the Domain Name System, i.e., the domain name in the URL functions as a namespace separator for the (local) identifier of a given resource. For URNs, a registry exists for various namespace identifiers that map to existing unique identifiers, e.g., for ISBNs, URNs can be created in in the form of `urn:isbn:<ISBN>`. National libraries have an own namespace identifier (`nbn`). The national namespace is then further divided. For example, every library in Germany can create unique URNs using the following naming scheme: `urn:nbn:de:<library-union>:<library-id>-<local-identifier><check-sum>`.

Blank Node: A blank node indicates that a resource exists, without using, or saying anything about, the identifier of this resource. This means that a blank node does not imply that an identifier in form of a URI exists for this resource.

Literal: Literals are used to identify values such as numbers and dates by means of a lexical representation. Two types of literals exist in RDF: plain literals and typed literals. A *plain literal* is a character string combined with an optional language tag, for example `"Eiffel Tower"@en`. A *typed literal* is a string combined with a datatype URI, for example `"324"^^xsd:integer` representing the integer value 324.

With these prerequisites, we now have everything to define statements in RDF in the form of RDF triples[2]:

Definition 2.4 (RDF Triple) *An RDF triple contains three components:*

- *the subject, which is an* RDF URI reference *or a* blank node,
- *the predicate, which is an* RDF URI reference,
- *the object, which is an* RDF URI reference, *a* literal *or a* blank node.

[2] http://www.w3.org/TR/rdf-concepts/#section-triples

An RDF triple is conventionally written in the order subject, predicate, object. The predicate is also known as the property of the triple.

To illustrate the use of RDF, we convert the statement from above: "The Eiffel tower has a height of 324 meters." By converting subject and predicate to RDF URI references and the object to a literal, we get the following RDF triple[3]:

```
ex:eiffeltower ex:height-in-meters "324"^^xsd:integer.
```

While the meaning of the statement is now formalized, we lost some additional information, like the name of the Eiffel tower. The fact that it is reflected in the URI is absolutely meaningless to a computer, we could also use ex:123 as URI for the tower. Therefore, we add this information and the information how the predicate can be expressed in a more human-friendly way by means of the predefined predicate `rdfs:label`:

```
ex:eiffeltower rdfs:label "Eiffel tower"@en.
ex:height-in-meters rdfs:label "has a height in meters of"@en.
```

This example shows a fundamental feature of RDF: the properties are resources just like any other resource, they are instances of the class `rdf:Property`. They can be used as subjects in statements and therefore be described by means of other properties. In particular, properties can also be related to other properties this way.

In this section, we so far more or less silently introduced already four specific resources – not including the example resources: `rdfs:Resource`, `xsd:integer`, `rdfs:label`, and `rdf:Property`. RDF and RDFS provide a set of resources that form a basic vocabulary to express commonly needed properties and simple logical assertions.

The property `rdf:type` assigns a class to a resource:

```
ex:eiffeltower rdf:type ex:tower.
```

[3] Throughout this document, we use the following namespace prefixes:
rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs=http://www.w3.org/2000/01/rdf-schema#
xsd=http://www.w3.org/2001/XMLSchema#
owl=http://www.w3.org/2002/07/owl#
ex=http://example.org/
dc=http://purl.org/dc/elements/1.1/
dcterms,dct,dc=http://purl.org/dc/terms/ (cf. Section 3.1)
dcam=http://purl.org/dc/dcam/
foaf=http://xmlns.com/foaf/0.1/
ore=http://www.openarchives.org/ore/terms/
edm=http://www.europeana.eu/schemas/edm/

This means that `ex:tower` is a class, which is an own type in RDF:

```
ex:tower rdf:type rdfs:Class.
```

Like properties, classes are resources that can be used as instances and therefore be described and related to other classes:

```
ex:tower rdfs:subClassOf ex:building.
```

Likewise, properties can be related to other properties:

```
ex:height-in-meters rdf:type rdf:Property.
ex:height-in-meters rdfs:subPropertyOf ex:height.
```

Besides the super class `rdfs:Resource`, the class of classes `rdfs:Class`, and the class of properties `rdf:Property`, we also have the class of literals `rdfs:Literal` and the class of data types `rdfs:Datatype` where `xsd:integer` belongs to. Not least, we can make statements about the *domain* and *range* of a property, i.e., about class memberships of resources that can be inferred from the fact that a resource is used in a statement as subject or object for a given predicate. From

```
ex:teaches rdfs:domain ex:Person.
ex:teaches rdfs:range ex:Lecture.
```

and the statement "`ex:kai ex:teaches ex:semantic-web`." follows:

```
ex:kai rdf:type ex:Person.
ex:semantic-web rdf:type ex:Lecture.
```

Domain and range statements have a far-reaching effect and should be made very carefully. In return, domain and range statements have to be taken into account when a property is to be applied in a new context. Neglecting them can lead to logical inconsistencies, as we will see in the remainder of this thesis.

2.2 Linked Data and the Semantic Web

RDF is only one side of the coin. Its full power is unleashed when it is used in the Web. To see why, we first have to understand that the RDF statements form a graph structure, when they are combined. Consider the statements that we used so far. They form an graph as illustrated in Figure 2.1.

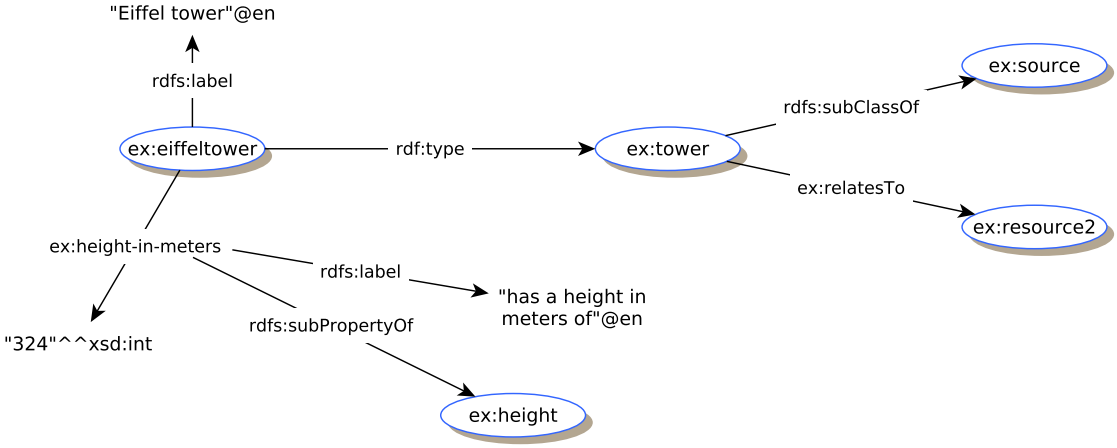

Figure 2.1: A graph of statements about the Eiffel tower

As the URIs used in this graph are dereferencable URLs, the Web can be used as infrastructure to provide information about the resources. When an application (or a human with a web browser) looks up such a URI, a description of the resource identified by the URI can be returned. There is only one problem: what if the URI identifies a resource that is actually a web resource (information resource), like a JPEG image. In this case, the image, i.e., the resource itself, should be returned, not a description. This is obviously not the case for other resources (non-information resources) like the Eiffel tower. We don't expect to retrieve the actual Eiffel tower when we look up its URI. To resolve these issues, the HTTP protocol has to be included.

If a non-information resource is requested, the server answers with a redirect (303) to another URI that contains metadata about the resource in RDF. If an information resource is requested, the server responds with the requested resource, except the requesting application asks specifically for an RDF description by means of a special request header (Accept).

This way, the Web can be used to connect all the RDF graphs by means of the URIs contained in the statements. A huge distributed database emerges, with countless links between the local RDF graphs. Or in other words: we get *Linked Data*. Due to the possibilities of RDF and OWL, an RDF based language to define ontologies, to describe and relate the resources semantically, further relations can be derived from the data automatically (by means of reasoning). The data becomes machine-interpretable, the Web turns into the *Semantic Web*. For the purpose of this thesis, the terms *Linked Data* and *Semantic Web* are interchangeable.

3 Provenance

I keep six honest serving-men:
(They taught me all I knew)
Their names are What and Why and When
And How and Where and Who.

Rudyard Kipling[1]

Provenance information is a special kind of metadata. As stated in Definition 1.1, "Provenance of a resource is a record that describes entities and processes involved in producing and delivering or otherwise influencing that resource." Provenance tracking is a cross-domain requirement with applications in software engineering (Davies, German, Godfrey, & Hindle, 2011), databases (Buneman, Khanna, & Wang-Chiew, 2001),(Green, Karvounarakis, & Tannen, 2007), (scientific) workflows (Davidson & Freire, 2008), and many others.

For the representation of provenance information, special vocabularies and data models have been developed, like the Open Provenance Model (OPM),[2] Provenir,[3] or the Provenance Vocabulary.[4] A good overview on provenance – also as a research topic – is provided by the W3C Provenance Incubator Group (2010) and Moreau (2010).

3.1 Dublin Core as a Simple Provenance Vocabulary

The Dublin Core Metadata Initiative[5] (DCMI) provides a core metadata vocabulary, commonly refered to as *Dublin Core*. Originally, it consisted of 15 elements that are

[1] From "Just So Stories", (Kipling, 1902).
[2] http://openprovenance.org/
[3] http://wiki.knoesis.org/index.php/Provenir_Ontology
[4] http://purl.org/net/provenance/
[5] http://www.dublincore.org

still available and called the *element set* (DCMI Usage Board, 2010b). The elements are defined very broadly, in particular they have no range specification, i.e., they can be used with arbitrary values as objects. The elements have been further refined and types have been introduced. This more specific vocabulary is called the *terms* and currently consists of 55 properties (DCMI Usage Board, 2010a).

The Dublin Core elements are considered legacy and the use of the DCMI terms is preferred. Both have different namespaces, usually the elements are used with the `dc` prefix, the terms with `dcterms` or `dct`. In this thesis, we are only concerned with the DCMI terms, so all prefixes refer to the terms if not explicitly stated otherwise.

Consider the following example for a metadata record:

```
ex:doc1 dct:title "A mapping from Dublin Core..." .
ex:doc1 dct:creator ex:kai .
ex:doc1 dct:created "2012-02-28" .
ex:doc1 dct:publisher ex:w3c .
ex:doc1 dct:issued "2012-02-29" .
ex:doc1 dct:subject ex:dublincore .
ex:doc1 dct:replaces ex:doc2 .
ex:doc1 dct:format "HTML" .
```

Clearly not all metadata statements deal with provenance. For instance, `dct:title`, `dct:subject` and `dct:format` are descriptions of the resource `ex:doc1`. They do not provide any information how the resource was created or modified in the past.

On the other hand, some statements imply provenance-related information, e.g., `dct:creator` implies that the document has been created and refers to the author. Similarly, the existence of the `dct:issued` date implies that the document has been published. This information is redundantly implied by the `dct:publisher` statement as well. Finally, `dct:replaces` relates our document to another document `ex:doc2` and it can be inferred that this document had probably some kind of influence on our document `ex:doc1`, which also gives us some provenance related information.

This is a pattern that applies generally to metadata, i.e., we can distinguish *description* metadata and *provenance* metadata. To be more precise, we define provenance metadata as metadata providing provenance information according to Definition 1.1 and description metadata as all other metadata.

Based on this definition, the DCMI terms can be classified as follows:

Description metadata: abstract, accessRights, accrualPeriodicity, accrualPolicy, alternative, audience, bibliographicCitation, conformsTo, coverage, description, educationLevel, extent, format, identifier, instructionalMethod, isRequiredBy, language, license, mediator, medium, relation, requires, rights, spatial, subject, tableOfContents, temporal, title, type.

Provenance metadata: accrualMethod, available, contributor, created, creator, date, dateAccepted, dateCopyrighted, dateSubmitted, hasFormat, hasPart, hasVersion, isFormatOf, isPartOf, isReferencedBy, isReplacedBy, issued, isVersionOf, modified, provenance, publisher, references, replaces, rightsHolder, source, valid.

There are 26 terms out of 55 that can be considered as provenance related. We will briefly discuss them based on the different aspects of provenance that are described here:

Who? (*contributor, creator, publisher, rightsHolder*) All properties have the range `dct:Agent`, i.e., a resource that acts or has the power to act. The *contributor*, *creator*, and *publisher* clearly influence the resource and therefore are important for its origin. This is not immediately clear for the *rightsHolder*, but as ownership is considered *the* important provenance information for artworks, it has to be included here.

When? (*available, created, date, dateAccepted, dateCopyrighted, dateSubmitted, issued, modified, valid*) Dates typically belong to the provenance record of a resource. It can be questioned if a resource changes by being published, however, we consider the publication as an action that affects the state of the resource and therefore it is relevant for the provenance. Two dates can be considered special regarding their relevance for provenance: *available* and *valid*. They are different from the other dates as by definition they can represent a date range. Often, the range of availability or validity of a resource is inhererent to the resource and known beforehand – consider the validity of a passport or a credit card or the availability of a limited special offer. In these cases, there is no action involved that makes the resource invalid or unavailable, it is simply determined by the validity range. On the other hand, if an action is involved, e.g., a resource is declared invalid because a mistake has been found, this is relevant for its provenance.

How? (*isVersionOf, hasVersion, isFormatOf, hasFormat, references, isReferencedBy, replaces, isReplacedBy, source, hasPart, isPartOf, accrualMethod*) Resources are

often derived from other resources. In this case, the original resource becomes part of the provenance record of the derived resource. Derivations can be further classified as *isVersionOf, isFormatOf, replaces, source*. *references* is a weaker relation, but it can be assumed that a referenced resource influenced the described resource and therefore it is relevant for its provenance. The respective inverse properties do not necessarily contribute to the provenance of the described resource, e.g., a resource is usually not directly affected by being referenced or by being used as a source – at most indirectly, as the validity state can change if a resource is replaced by a new version. However, inverse properties belong to the provenance related terms as they can be used to describe the relations between the resources involved. The last three properties are special as they are specific for collections of resources. We intentionally skipped the order here, as hasPart is in this case the property that contributes to the provenance of the described resource, while isPartOf is the inverse.

This leaves one very special term: *provenance*. It is defined as a "statement of any changes in ownership and custody of the resource since its creation that are significant for its authenticity, integrity, and interpretation." This refers again to the traditional definition of provenance for artworks. This nicely illustrates how this definition relates to the one that we use in this thesis. Actually, almost half of all DCMI terms tell us more about the provenance of a resource than this dedicated term. Nevertheless, it is of course relevant for provenance.

In summary, the DCMI terms – and therefore any Dublin Core metadata record – hold a lot of provenance information and tell us about a resource, *when* it was affected in the past, *who* affected it and *how* it was affected. What about the other questions? The description metadata, i.e., the other DCMI terms, tells us *what* was affected. Indeed there are no direct information in Dublin Core, *where* a resource was affected. Such information is usually only available for the publication of a resource, i.e., this action is located at the address of the publisher. Note that *spatial* is not related to this question, as this is a descriptive property that tells us for instance that a book is about Berlin, but not that it was created in Berlin – or even that it has ever been or is otherwise related to Berlin. And finally, the question, *why* a resource was affected, lacks – apart from subtle hints from terms like *replaces* – as usual a satisfying answer.

3.2 W3C Provenance Working Group

The W3C Provenance Working group currently develops specifications for the interoperable interchange of provenance information in heterogeneous environments such as the Web (W3C Provenance Working Group, 2012b). The targeted date for a published recommendation is January 2013. This means that all information in this section have to taken with care, as it can only reflect the current work in progress. We therefore try to omit the details and instead provide the bigger picture.

The family of specifications developed by the working group is called PROV.[6] PROV consists of several specifications, in particular the PROV data model (PROV-DM) and the PROV ontology (PROV-O). Both are described in working drafts (W3C Provenance Working Group, 2012a, 2012c). PROV-DM is written in a formal language, PROV-O translates PROV-DM to RDF using the OWL2 Web Ontology Language (W3C OWL Working Group, 2009), an RDF based language to define ontologies. As we are concerned with the Semantic Web in this thesis, we only refer to PROV-O.

The main aspect that distinguishes PROV – and other provenance models like OPM – from Dublin Core is that it is based on activities that affect the described resource. This means that an agent is not directly related to a resource, like with `dct:creator`, instead, the agent is associated with an activity that leads to the creation of the resource. The following example taken from (W3C Provenance Working Group, 2012c) illustrates this in PROV-O.

It describes part of the provenance of a bar chart (`ex:bar_chart`) that was created by Derek (`ex:derek;`). The creation was an activity (`ex:illustrationActivity`). The bar chart was generated (`prov:wasGeneratedBy`) by this activity, the activity was associated with (`prov:wasAssociatedWith`) Derek. For this activity, a dataset (`ex:aggregatedByRegions`) was used (`prov:used`), which means that the bar chart was derived (`prov:wasDerivedFrom`) from this dataset.

```
ex:bar_chart[7]
     a prov:Entity;
     prov:wasGeneratedBy ex:illustrationActivity;
     prov:wasDerivedFrom ex:aggregatedByRegions.
```

[6] In the charter and some earlier drafts, it was refererenced as Provenance Interchange Language (PIL)

[7] These examples are provided in Turtle, the Terse RDF Triple Language (Beckett & Berners-Lee, 2011). `a` is a shorthand for `rdf:type`, several statements with the same subject are combined using "`;`" as separator, multiple objects can be assigned at once by simply listing them with "`,`" as separator.

```
ex:illustrationActivity
     a prov:Activity;
     prov:used ex:aggregatedByRegions;
     prov:wasAssociatedWith ex:derek.
```

This is only the first part of the full example, in (W3C Provenance Working Group, 2012c) a full provenance chain to the original data source is provided. However, this minimal example should be sufficient to get the idea.

In the example, you can see that there is the indirect relation between the bar chart and the dataset via the activity, but also a direct relation stating that the bar chart was derived from the dataset. There are more such "shortcuts" in PROV-O, e.g., the property `prov:wasAttributedTo` that assigns an agent directly with a given entity, which is more similar to the `dct:creator` property in Dublin Core.

A distinctive feature in PROV is the definition of the class `prov:Entity`. The current definition in PROV-DM (W3C Provenance Working Group, 2012a) reads as follows: "Things we want to describe the provenance of are called entities in PROV. An entity is a physical, digital, conceptual, or other kind of thing; entities may be real or imaginary." So far this corresponds to the definition of a resource as we used it (Definition 2.2). However, it is further stated about activties: "Generation is the completion of production of a new entity by an activity. This entity did not exist before generation and becomes available for usage after this generation." This is a useful definition, as PROV aims at the representation of a full provenance chain and this can only work, if the different *states* of a resource during its creation and lifecycle can be identified and distinguished. However, this means that there can and typically will be several entities that belong to one resource. Consider for instance a constantly modified web page, like a wiki page. There is a URI for the wiki page, i.e., the wiki page is a resource (and an entity), e.g, `http://example.org/wiki/page1`. Every modification of the wiki page however leads to a new entity (and a resource) with an own URI, e.g., `http://example.org/wiki/page1?rev=3`. In PROV, these entities can be related among themselves by `prov:alternateOf` and to the entity of the wiki page by `prov:specializationOf`.

PROV provides further qualifications for the relations between entities, agents, and activities. For instance, the association of Derek in the example above could be further qualified by stating that he had a certain role in this activity as team leader:

```
ex:illustrationActivity
     prov:qualifiedAssociation [8
          a prov:Association;
          prov:agent ex:derek;
          prov:hadRole ex:teamleader;
     ].
```

Likewise, the usage, generation, and derivation can be qualified.

This is only an extract from PROV, there are many more properties and specialized classes. But the given examples should be sufficient to provide an idea how PROV compares to Dublin Core and how the representation of full provenance information in RDF could look like. In summary, PROV is activity based, i.e., all activities that affected the state of a resource are described and thus a provenance chain of entities and agents is created that can be traced back to the origin. This makes PROV especially useful, if different versions of a resource and the whole lifecycle has to be tracked. The drawback is a higher complexity compared to the simple way how basic provenance information is provided by means of Dublin Core.

3.3 Mapping from Dublin Core to PROV

Regarding the information provided in this section, the same restrictions apply as in the last section. This mapping is developed as part of the W3C Provenance Working Group activity and only briefly introduced based on its current state.

Why are we concerned with a mapping between Dublin Core and PROV? First, such a mapping can provide valuable insights into the different characteristics of both data models, in particular it "explains" PROV from a Dublin Core view point. Second, such a mapping can be used to extract PROV data from the huge amount of Dublin Core data that is available on the Web today. Third, it can translate PROV data to Dublin Core and therefore make it accessible for applications that understand Dublin Core. And not least, it can lower the barrier to adopt PROV, as simple Dublin Core statements can be used as starting point to generate PROV data.

[8] Another shortcut in Turtle: "`[]`" represents a blank node, the contained statements describe this blank node, i.e., they have the blank node as subject.

The development of this mapping is a joint activity of the DCMI Metadata Provenance Task Group and the W3C Provenance Working Group and will become part of the PROV documentation.

3.3.1 Basic considerations

Substantially, a complete mapping from Dublin Core to PROV consists of three parts:

1. Direct mappings between terms that can be expressed in form of subclass or subproperty relationships in RDFS – or equivalent relationships in OWL.
2. Definition of new specializations (subclasses or subproperties) of the target vocabulary to reflect the expressiveness of the source vocabulary.
3. Provision of complex mappings that create statements in the target vocabulary based on statements in the source vocabulary.

In this thesis, we only consider one direction, Dublin Core to PROV. For the complex mappings, we take the following approach:

Stage 1. In the first stage, only single DC statements are mapped to PROV. Relations between several statements affecting the resulting PROV statements are not yet taken into account. For example, if a specialization of a document is generated by one activity and a specialization is used by a different activity later in time, it can be assumed that both are the same entities, if the second activity directly follows the first activity. These conflations and other clean-up steps are performed in the second stage. A rationale for these two steps is that the mappings in stage 1 are context free and do not depend on the existence of any other statements. On the other hand, by employing the patterns developed for stage 2, any kind of generated PROV data could be cleaned up at a later point, for instance after the integration with provenance information from a different source, which could be advantageous.

Stage 2. In the second stage, we employ reasoning patterns to clean-up the data, e.g. by conflating blank nodes that are actually the same or by identifying a final specialization of the original document that is identical to this document – see question below: "What is `ex:doc1`?" At the time of this writing, we develop stage 1, therefore we cannot provide more details about stage 2 at this point apart from these general ideas.

3.3.2 What is `ex:doc1`?

Consider the example metadata record in Section 3.1. As a DC metadata record describes the resulting document as a whole, it is not clear, how this document relates to the different states that the document had until it reached its final state. For example, a document can have assigned a `dct:created` date and a `dct:issued` date. The activity of issuing a document does not necessarily change the document, but regarding the PROV ontology, there are two different specializations of this document before and after the issuing activity, distinguishable by the property of the document that states if the document was issued. Generally, there are two possibilities how to deal with this:

1. We can always create new instances of entities, typically as blank nodes, that all are related to the original document by means of `prov:isSpecializationOf`. This leads to bloated and not very intuitive data models, e.g. think about the translation of a single `dct:creator` statement, where you would expect to somehow find some activity and agent that are directly related to the document (cf. Figure 3.1).

2. We can always use the original document as the instance that is used as `prov:Entity`. The implications regarding the semantics of a `prov:Activity` are not yet totally clear, however, it contradicts the above mentioned definition to have an activity that uses an entity and generates the same entity. If an activity actually generates an entity, it is semantically incorrect to have several activities that all generate the same entity at different points in time.

As the first option is the more conservative one with respect to the underlying semantics, our proposal is to use it in the first stage. This still leaves interesting questions for the second stage:

1. How do we reduce the number of specializations, e.g., by stating that the specialization that is generated by activity 1 is the same entity that is used by activity 2?

2. How do we relate the specializations to `ex:doc1`? We could create two entities based on the actual creation activity: `ex:doc1` and a first specialization. We could further declare the last produced specialization as the same entity as `ex:doc1`. Depending on the underlying data, this can be the entity that is identified by the URI of the original document. However, we have to be careful to avoid cycles in the provenance we produce. For now, this remains undecided.

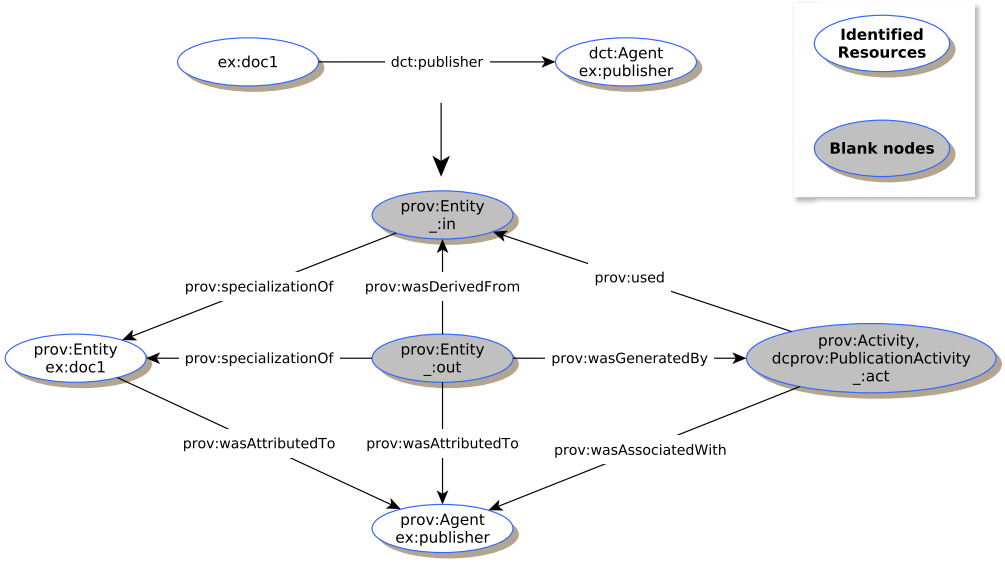

Figure 3.1: A possible mapping from Dublin Core to PROV

3.3.3 Direct Mappings

Direct mappings can particularly be provided for classes and the "shortcuts", i.e. the direct relationships in PROV between an entity and an agent or an entity and a date. Examples for direct mappings would be:

```
dct:references rdfs:subPropertyOf prov:wasDerivedFrom .
dct:creator rdfs:subPropertyOf prov:wasAttributedTo .
dct:rightsHolder rdfs:subPropertyOf prov:wasAttributedTo .
dct:date rdfs:subPropertyOf prov:generatedAtTime .
dct:Agent owl:equivalentClass prov:Agent .
prov:hadOriginalSource rdfs:subPropertyOf dct:source .
prov:wasRevisionOf rdfs:subPropertyOf dct:isVersionOf .
```

The direct mappings are straight-forward and provide basic interoperability using the integration mechanisms of RDF. By means of RDFS-reasoning, any PROV application can at least make some sense from Dublin Core this way. The direct mappings also contribute to the formal definition of the vocabularies by translating them to the other vocabulary. Interestingly, it can be seen from these few examples, that Dublin Core, while less complex from a modeling perspective, is more specific about the type of the activity. PROV only provides general attribution, the details about the kind of influence

that an activity or an agent had are left to custom specializations of the PROV classes and properties.

3.3.4 PROV Specializations

This leads us to the next part. To properly reflect the meaning of the Dublin Core terms, we need such specializations, like the following:

```
dcprov[9]:CreationActivity rdfs:subClassOf
     prov:Activity, dcprov:ContributionActivity .
dcprov:ContributionActivity rdfs:subClassOf
     prov:Activity .
dcprov:CreatorRole rdfs:subClassOf
     prov:Role, dcprov:ContributorRole .
dcprov:ContributorRole rdfs:subClassOf
     prov:Role .
```

Custom specializations of the properties should be omitted as they would be identical to the Dublin Core terms. If these specializations are wanted, the Dublin Core terms should be used directly, according to the specialization relations above.

3.3.5 Complex Mappings, Stage 1

The complex mappings are provided in form of SPARQL CONSTRUCT queries, i.e., queries that describe a resulting RDF graph based on another RDF graph found in the original data. Based on the categorization of the terms in Section 3.1, we provide three examples:

Entity/Agent (Who). In this category, we have four terms: *contributor*, *creator*, *publisher*, and *rightsHolder*. The former three can be mapped with the same pattern, as follows, only the roles and activities change and for publication, a second specialization would be created that represents the entity before the publication (see below):

[9] This namespace prefix is unassigned, as these classes are currently only proposals.

```
CONSTRUCT {
     ?doc a prov:Entity .
          prov:wasAttributedTo ?ag .
     _:out a prov:Entity .
          prov:specializationOf ?doc .
     ?ag a prov:Agent .
     _:act a prov:Activity, dcprov:CreationActivity ;
          prov:wasAssociatedWith ?ag ;
          prov:qualifiedAssociation _:assoc .
     _:assoc a prov:Association ;
          prov:agent ?ag ;
          prov:hadRole dcprov:CreatorRole .
     _:out prov:wasGeneratedBy _:act ;
          prov:wasAttributedTo ?ag .
} WHERE {
     ?doc dct:creator ?ag .
}
```

In this query, `?doc` and `?ag` are variables that are set to different matching values depending on the data found in the triple store. The graph in the CONSTRUCT part can be seen as a template where the variables are placeholders that are filled with the values found in the data. The mapping corresponds to the graph in Figure 3.1, where only the qualified association is omitted. Therefore Figure 3.1 contains a second entity that is used by the activity. This entity is missing here as we assume that no entity exists before the creation activity. With this mapping, the difference in the complexity becomes obvious. A lot of blank nodes are created, however, keep in mind that we envision a second stage that relates them and provides stable URIs for the entities.

Entity/Date (When). The dates often correspond with a who-property, e.g., *creator* and *created* or *publisher* and *issued*. Therefore, they lead to similar statements, only providing a date instead of an agent associated with the activity. We use *issued* as an example here, because from issued, two specializations can be inferred: something must be available before it can be published.

```
CONSTRUCT {
     ?doc a prov:Entity .
     _:act a prov:Activity ;
          prov:used _:in .
     _:out a prov:Entity ;
          prov:specializationOf ?doc1;
          prov:wasGeneratedBy _:act;
          prov:qualifiedGeneration _:gen;
          prov:wasDerivedFrom _:in .
          prov:generatedAtTime ?date .
     _:gen a prov:Generation ;
          prov:atTime ?date ;
          prov:entity _:out .
     _:in a prov:Entity ;
          prov:specializationOf ?doc1 .
} WHERE {
     ?doc dct:issued ?date.
}
```

Entity/Entity (How). Most Dublin Core terms in this category are related to `prov:wasDerivedFrom`. They can be mapped directly, but also a complex mapping can be provided. Here, a specialty of SPARQL CONSTRUCT queries can be used to deal with the inverse properties in Dublin Core:

```
CONSTRUCT {
     ?doc1 a prov:Entity ;
          prov:wasDerivedFrom ?doc2.
     ?doc2 a prov:Entity .
} WHERE {
     OPTIONAL { ?doc1 dct:isVersionOf ?doc2 .  }
     OPTIONAL { ?doc2 dct:hasVersion ?doc1 .}
}
```

The `OPTIONAL` keyword means that the included statement does not need to exist. Triples in the resulting graph with variables that have no binding simply are omitted. In

this case this leads to the correct PROV statement, if either or both source statements are present.

From the entity/entity relations, an activity can and should also be inferred. We omit it here for brevity.

In essence, these examples sketch the first stage of the mapping. As everything is provided as RDF statements or SPARQL CONSTRUCT queries, this mapping can simply applied to arbitrary RDF data by adding the statements and the resulting graphs from the queries to the data. Even without the clean-up from stage-2, these steps already lead to a lot of PROV data that can be reused by provenance-aware applications.

3.4 Conclusion

In this chapter, we provided the necessary foundations regarding the representation of provenance information in the Semantic Web. We introduced Dublin Core as a simple provenance vocabulary and recapitulated the current work in the W3C Provenance Working Group and the DCMI Metadata Provenance Task Group regarding the development of the PROV specifications and the mapping between Dublin Core and PROV. The mapping demonstrates, how much provenance information is "hidden" in Dublin Core metadata, which is not surprising considering the fact that almost half of the Dublin Core terms are related to provenance information.

Based on these considerations it is possible to recommend a vocabulary depending on the desired use case. If simple and intuitive metadata is wanted that represents the most important and basic facts on who created a resource, when it was created and how it relates to other resources, then Dublin Core is recommended. If the whole provenance chain of a resource has to be tracked, possibly with additional information about the underlying workflow and the lifecycle of the resource, then PROV provides a comprehensive framework.

Due to the provided mapping, it can be expected that Semantic Web applications can make sense of both models, if they understand at least one.

4 Metadata Provenance

> *Metadata, you see, is really a love note – it might be to yourself, but in fact it's a love note to the person after you, or the machine after you, where you've saved someone that amount of time to find something by telling them what this thing is.*
>
> Jason Scott[1]

Now that we know what provenance metadata is and how it can be represented in RDF, we want ro restrict the domain for which provenance metadata is provided. In this chapter, we investigate difficulties and possible approaches how to represent the provenance *of* metadata.

4.1 Metadata Characteristics

At first sight, it is not clear why the representation of provenance information for metadata should be special or different from representations of provenance for other, arbitrary resources. This question, however, already pinpoints the main problem: metadata is often not seen as a resource of its own. It is just "there" describing other resources. We identify the following typical factors that support this notion:

Metadata arises from applications: Metadata is created and managed in applications dealing with resources. In the context of a typical application, the representation of any information about the metadata is not needed. Consider for example a file system that creates metadata like creation and modification dates, size, and owner for any file stored in the system. Why would a developer of a file system add a further level to make the metadata describable? Any database application can be

[1] http://ascii.textfiles.com/archives/3181

seen as a metadata application: a customer database contains (meta)data about customers, a library catalog contains metadata about books. The fundamental split of the world in two levels seems natural: the level of resources and the level of their descriptions. Fundamental means that this split is inherent to the applications; adding the possibility to describe the descriptions would strongly affect the database design and software architecture.

Metadata is tied to resources: To decouple metadata from the applications, mechanisms exist to represent a resource together with its description. For example, PDF, JPEG or MP3 files can contain metadata about the creator of a document, picture or song. Again, the two levels remain naturally: The resource and its contained description.

Sloppy introduction of metadata provenance: Despite the distinction of the two fundamental levels, sometimes, metadata provenance is needed. But then it is often added on a case-by-case basis and *mixed* with the metadata. Think of a last-modified column in a database application. While all other columns in a customer table describe the customer, this column obviously does not. Instead, it describes the table row and denotes when this row (and not the customer) was modified last. This imprecision in the distinction of description and described resource is common and can also be explained by the deeply anchored perception of the two levels: why should we be more precise, isn't it clear what belongs to the description and what to the described resource?

Synonymous use of description and described resource: Within a system the description of a resource often *is* the resource. In library catalogs, metadata records about books have identifiers and it is tempting to use them as identifiers for the described books. There is nothing to be said against a consistent use of the identifiers in this way, as long as they don't function at the same time as identifiers for the records. The problem is that this is often neglected and decided on a case-by-case basis, which leads to the sloppy provenance as described above. The only solution is the introduction of two identifiers, one for each of the two fundamental levels. Only then, their distinction can be ensured consistently.

In short: to talk about metadata, e.g., to give its provenance, the metadata needs to be decoupled from the resource *and* the metadata needs to be identifiable. Then it becomes distinguishable from the resource. Moreover, it becomes a resource on its own. Figure 4.1 illustrates this process. First, resources and metadata are mixed together.

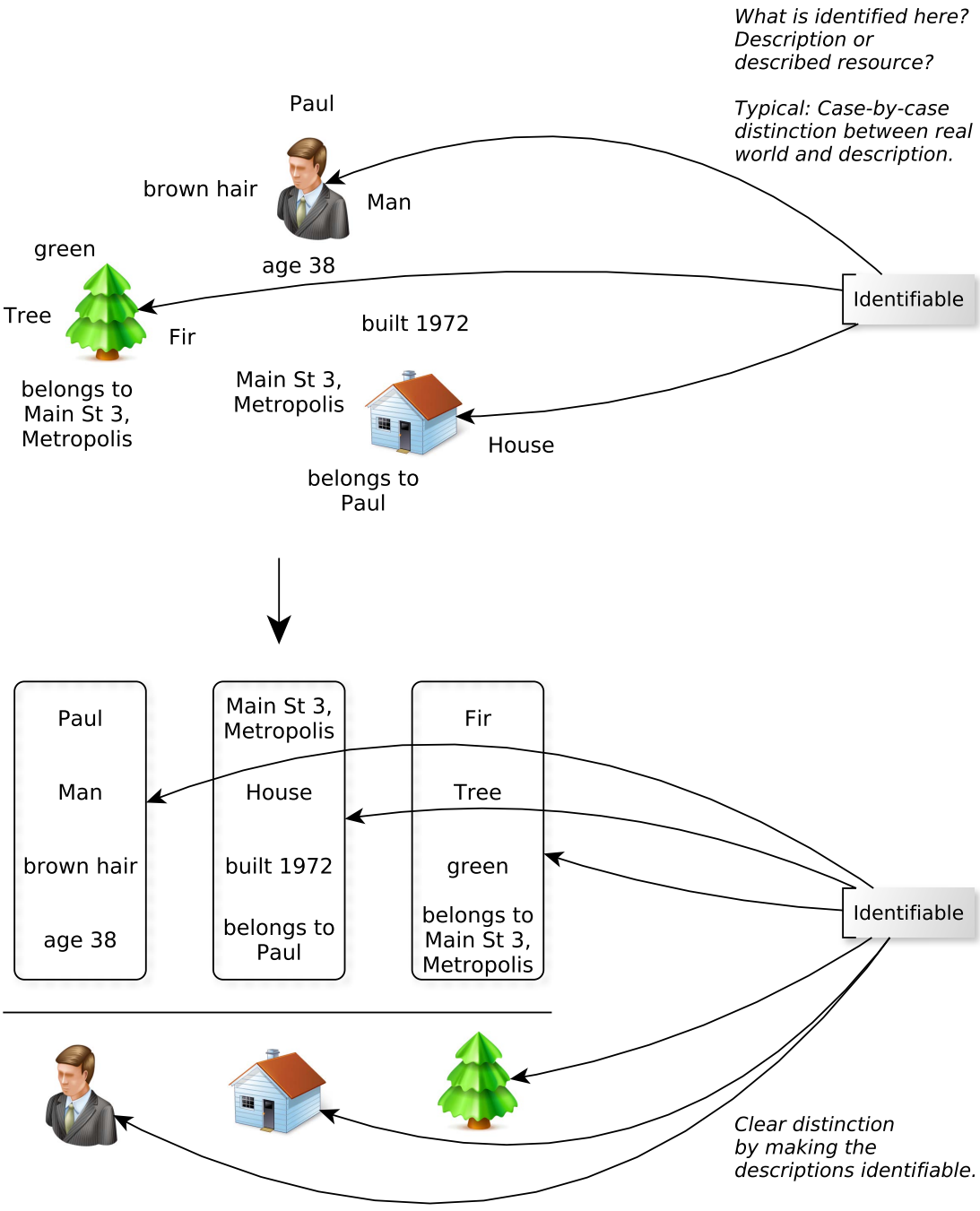

Figure 4.1: From descriptions to identifiable resources

Note that they *are* distinguishable, with images representing the resources and text representing the metadata. This corresponds to the intuitive distinction of resources and metadata that we do all the time, in the perception of the world, as well as in our applications. By making the metadata identifiable, we move from the intuitive to an explicit distinction.

4.2 Metametadata in RDF

Decoupling of metadata from the described resources is accomplished with RDF. Even with embedded forms of RDF like RDFa[2] the metadata is clearly distinguishable by its formalized representation. Of course, with RDF it is still possible to mix different levels of abstraction, e.g., by assigning the `dcterms:date` 1889 to a webpage about the Eiffel tower. It is, however, arguably less likely as one is forced to think about the meaning of a resource when it is referenced. Using a URI reference both for the Eiffel tower and a webpage about it is not bad practice, it is just wrong.

Not accomplished is the identification of metadata in RDF. There are various approaches and best-practices, but yet a satisfying standardized solution is missing. In this section, we aim at giving an account of the available approaches and the developments that hopefully lead to a solution in the upcoming next version of RDF. All approaches have in common that they make RDF triples or sets of RDF triples identifiable as resources; only then it is possible to describe them. We will see that this shift to the next metalevel has far-ranging implications for RDF. In (Eckert, Pfeffer, & Stuckenschmidt, 2009), we coined the term "Metametadata" to indicate the distinctiveness of such approaches from the mere provision of metadata for arbitrary resources.

4.2.1 Linked Metadata

The first approach that has to be mentioned simply uses the linked data principles (W3C SWEO Interest Group, 2008; Heath & Bizer, 2011) to provide metadata about metadata. Whenever metadata is published on the web, a URI has to be coined that makes it accessible and identifiable, just like any other resource. Consider metadata about the Eiffel tower `ex:eiffeltower` (Figure 4.2). When `ex:eiffeltower` is dereferenced, the server 303-redirects to a URL where metadata about this non-information resource

[2] http://www.w3.org/TR/xhtml-rdfa-primer/

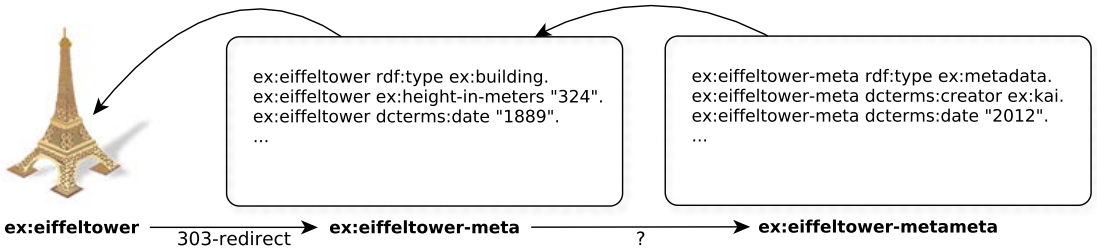

Figure 4.2: Linked Metadata

can be found, e.g., `ex:eiffeltower-meta`. `ex:eiffeltower-meta` is an information resource that can itself further be described, either with RDF statements that are directly delivered with the 200-response for `ex:eiffeltower-meta` or from a new URI, e.g., `ex:eiffeltower-metameta`.

In the former case, it would not be possible to further describe the metametadata, as it is not identified by an own URI. In the latter case, the question remains, how the metametadata can be retrieved. A general 303-redirect is not possible, as an application would like to retrieve the original metadata from `ex:eiffeltower-meta`. The typical practice for other resources is content-negotiation based on the HTTP accept header: the application determines for a resource – e.g., a JPEG image – if it would like to get the resource itself by indicating that the content-type JPEG is accepted or if it would prefer a description of the resource by indicating that the content-type RDF is accepted. This does not work here, as both the resource and its description have the content-type RDF.

A possible solution would be the introduction of a new HTTP request header indicating that metadata is desired. However, the counter-part for the response exists: the server can send a link header, indicating that metadata for the requested resource is available at a different URL:

```
Link:  <http://example.org/eiffeltower-metameta>; rel=meta
```

The recommended approach therefore is to ask for the headers of a resource by means of a head request and see, if a link header points to some metadata. Another approach that works for RDF data would be to refer to the description of the data within the RDF triples that are provided as resource. E.g., one could add the following statement: `ex:eiffeltower-meta rdfs:seeAlso ex:eiffeltower-metameta`. Unfortunately, there is no commonly accepted property that can be used for this purpose –

another possibility would be `foaf:page`. `rdfs:seeAlso` is very general for this quite fundamental and specific purpose.

These basic considerations already lead us to some problems of metadata provenance in RDF. It is clear that metadata provenance needs to fit seamlessly within the linked data environment and it can be expected that the approach as presented here will remain valid, as it just employs the linked data principles, irrespective of the type of the resource. There are, however, some drawbacks that make this approach not universally applicable:

1. It is only suitable on a coarse-grained level, as a request is needed for every identifiable subset of RDF triples. There are requirements – for instance raised by Hillmann, Dushay, and Phipps (2004) – for provenance information on statement level.
2. It requires the modification of response headers and the processing of request headers, i.e., full control over the web server is needed.
3. There are several degrees of freedom how to implement it, e.g., if and which headers or properties should be used and if the metametadata is provided on its own or together with the metadata.
4. It is not clear, how the metametadata is represented and organized within RDF applications. At least the information needs to be stored, via which URL a triple has been retrieved. But even then, the information from link headers is lost and the connection between metadata resources can only be inferred from their contents.

4.2.2 Reification

Since the first version of RDF exists a mechanism to express metalevel information on statement level: Reification. In (Eckert et al., 2009), we demonstrated how reification can be used generally to express provenance information.

Reification allows to describe single RDF statements. Therefore, RDF statements have to become resources. An RDF statement is an instance of the class `rdf:Statement`. The statement is defined by three properties: `rdf:subject`, `rdf:predicate`, and `rdf:object`. The reification of the statement "`ex:eiffeltower ex:height-in-meters "324"^^xsd:integer.`" looks like this:

```
ex:stmt1 rdf:type rdf:Statement.
ex:stmt1 rdf:subject ex:eiffeltower.
```

```
ex:stmt1 rdf:predicate ex:height-in-meters.
ex:stmt1 rdf:object "324"^^xsd:integer.
```

Now we can make further statements about this statement, e.g., we can indicate who created this statement: "`ex:stmt1 dcterms:creator ex:kai.`"

Despite the general applicability for the representation of metadata provenance on statement level, reification is not widely used and even its deprecation in the next version of RDF is considered (Hawke, 2011). Among the reasons is the cumbersome representation of a statement that leads to a triple explosion: four statements are needed to reify one statement – four additional statements, as the reification of a statement does not make the statement in RDF. The actual statement "`ex:eiffeltower ex:height-in-meters "324"^^xsd:integer.`" still has to be added. Moreover, every single statement has to be reified and somehow related to other triples, if a set of triples should be described. Svensson (2011) uses OAI-ORE (see Section 4.2.5) for this purpose to aggregate reified statements and therefore to represent a metadata record.

Reification is currently the only way to talk about single RDF statements within RDF. Other approaches, however, are currently developed and already have a much higher acceptance in the community, like the *Named Graphs* that can be seen as a groundwork for the next RDF version and that are described in the following.

4.2.3 Named Graphs

Carroll, Bizer, Hayes, and Stickler (2005) introduced named graphs as a minor extension of RDF. They developed named graphs from the notion of URI references for RDF files, as described above. While named graphs are not part of RDF, they have been included into SPARQL (RDF Data Access Working Group, 2008), the query language for RDF. A named graph is simply an RDF graph associated with a URI functioning as the name for the RDF graph. The name can be the URI reference of the RDF file where the graph is stored or provided in a different way. For instance, TriG (Bizer & Cyganiak, 2007) is a syntax to serialize Named Graphs.

In TriG, the example graph from above looks like this:

```
ex:eiffeltower-meta {
     ex:eiffeltower rdf:type ex:building.
     ex:eiffeltower ex:height-in-meters "324".
```

```
        ex:eiffeltower dcterms:date "1889".
        ...
}
```

Due to the inclusion in SPARQL, most implementations today support named graphs. From an implementation perspective named graphs are related to quads, i.e., the general extension of the RDF triples to quadruples, as proposed for instance by MacGregor and Ko (2003). If the fourth element in the quad is used for the name of the containing named graph, triples occurring in more than one graph have to be stored multiple times. Another concern is that a named graph has to be created for each statement, if metametadata on statement level has to be provided. For the purpose of this thesis such implementational details are not important, however, it should be mentioned that the meaning of the fourth element in a quad and the requirement for a (possibly additional) statement identifier is an ongoing discussion, for instance see (Ferris & Cyganiak, 2011).

In (Eckert, Pfeffer, & Völker, 2010), we used named graphs to demonstrate their applicability for our metametadata use cases as introduced in (Eckert et al., 2009). Both reification and named graphs are suitable means to make metadata statements or sets of statements identifiable. Named graphs are more intuitive for metadata applications as they are in line with the notion of metadata records. However, they lack standardization and are not part of the RDF core. Reification is, but its usage is cumbersome and at least a construct would be required to collect statements to sets of statements for provenance representation in typical metadata applications.

4.2.4 Recent developments: G-boxes, Layers, Surfaces, Bundles...

The requirement for such mechanisms have also been formulated by other researchers, for instance Zhao, Bizer, Gil, Missier, and Sahoo (2010). Currently, the W3C RDF Working Group works on the next version of RDF and the development of a standardized mechanism for graph identification is part of their charter.[3] One of the first things that the group did was the definition of intermediate terms to enable a proper discussion of the involved concepts. In particular, the following terms have been coined[4]:

g-box: A *g-box* is a container, like a *set* data structure in programming. It holds some RDF triples. G-boxes can overlap, i.e., they can contain identical triples. Two g-boxes can happen to have the same content while being distinct g-boxes. The

[3] http://www.w3.org/2011/01/rdf-wg-charter
[4] http://www.w3.org/2011/rdf-wg/wiki/GraphConceptTerminology

content of a g-box can change: today a particular g-box might contain the triples `{ my:a my:b _:x. my:a my:c _:x }` , and tomorrow it might instead contain `{ my:a my:b _:x. my:a my:c2 _:x }` .

g-snap A *g-snap* is an idealized snapshot of a g-box; it is a mathematical set of RDF triples. Like g-boxes, g-snaps can overlap, sharing triples. Unlike g-boxes, it makes no sense to talk about g-snaps changing: they are defined to be exactly the collection of their elements. If a g-snap were to "change" it would simply be a different g-snap. If two g-snaps have the same triples, they are really the same g-snap. The contents of a g-box at any point in time are a g-snap.

g-text A *g-text* is a particular sequence of characters or bytes which conveys a particular g-snap in some language, e.g., Turtle or RDF/XML. If you can parse a g-text, you know what is in the g-snap it conveys. You can provide the exact content of a particular g-box at some instant by providing a g-text – the g-text conveys the g-snap which is the current state/content of the g-box.

With these intermediate terms, we can now see that the identification of metadata can happen on different levels, with different implications for the implementation. An identified g-box equates roughly to a metadata resource on the Web or to a named graph. Therefore, this is the level that we have in mind in this thesis. However, the identification of a g-snap is also relevant for metadata provenance, as this would ensure that we are talking about a concrete state of the metadata. Nevertheless, the changeability of resources is common and therefore, mechanisms to identify a concrete state can generally be left to the implementors. In this respect, g-boxes behave just like any other resource.

There are a lot of discussions going on which term will be used later instead of g-box. Current proposals include *layer* and *surface*, both with slightly different conceptual ideas behind them. Both have their merits, we will see what finally becomes accepted.

In PROV, there is also a concept of a provenance record, probably called *bundle* – a former name was *account*. The W3C Provenance WG and the W3C RDF WG work closely together to ensure that ultimately the concept of a *bundle* fits to the concept of a *g-box*.

4.2.5 OAI-ORE

The lack of a standardized mechanism to express metalevel information and the shortcomings of reification require developers to find other solutions how to deal with the

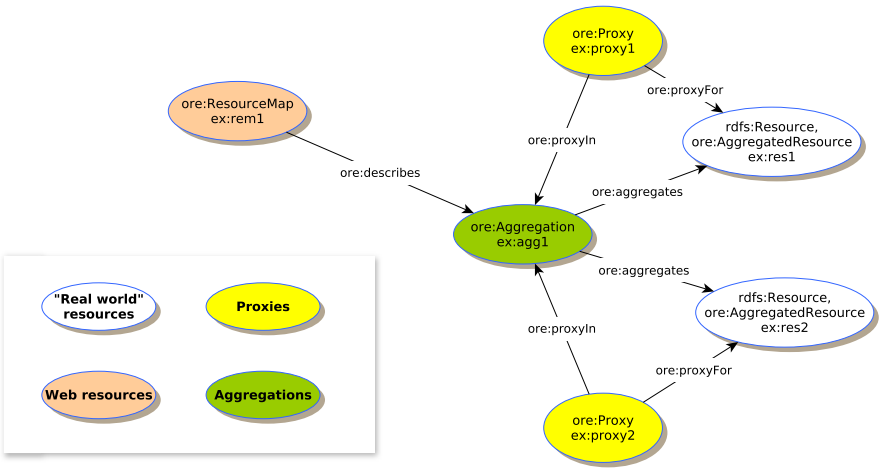

Figure 4.3: OAI-ORE Example

representation of metametadata. One approach that does not need the metalevel in RDF is the use of the *Object Reuse and Exchange* (ORE) framework provided by the Open Archives Initiative (OAI) (Open Archives Initiative, 2008b). OAI-ORE was originally developed to address a slightly different, yet very interesting problem, that also lacks a solution in RDF: how to make statements about resources that are only valid in a special context. A typical example woule be an aggregation of resources, e.g., a collection of articles that are bundled together. The statement that an article is the first article in this collection is only valid for this collection, the article is not generally the first article.

Figure 4.3 illustrates the data model. The approach taken by ORE is the introduction of proxies (`ore:Proxy`). A proxy is a special resource that represents the original resource (`ore:AggregatedResource`) within an aggregation (`ore:Aggregation`). Statements about the proxy are statements about the original resource, however, they only apply for the resource in the context of the aggregation.[5] Statements about the aggregation and its connections to aggregated resources and proxies are provided in a special resource, the resource map (`ore:ResourceMap`).

The OAI-ORE model is used in the Europeana Data Model to represent metadata provenance. We will see in Chapter 5 that the use of proxies can be problematic. Their general drawback is that their semantics differs from the standard semantics in RDF

[5]In ORE, proxies are optional, resources can also be aggregated directly. However, proxies are required if ORE is used to represent metametadata. Proxies are the means to introduce the metalevel – cf. (Svensson, 2011), who uses ORE without proxies, but therefore needs reification, as stated above.

where a statement generally is made about a resource directly. Applications have to "understand" ORE to make sense from statements about proxies. This is acknowledged in the documentation: it requires the HTTP server to handle applications that are not "ORE aware" differently and to 303-redirect them to the aggregated resource.

Apart from the proxies, OAI-ORE is a straight-forward approach to describe aggregations of resources. The resource maps simply use linked data practices, as described above, i.e., ORE only makes the relation between non-information resources (the aggregations) and their description explicit and requires the description to have a URI.

4.3 Dublin Core Abstract Model

In the last section, we investigated several ways to make metadata identifiable, the main requirement to talk about metadata like about any other resource. Based on the recent developments in the working groups, it can be assumed that there will be a satisfying and hopefully well-accepted way to do so in the near future.

However, this is only the implementational part, the technical foundation to even allow making statements about RDF data within RDF. It can be questioned if we need anything further, as in the end, all RDF data is metadata and identifying metadata is all we asked for. In this section, we propose an extension of the Dublin Core Abstract Model and its use as an RDF vocabulary to talk about metadata. This way we would like to make the notion of a set of statements belonging together as a metadata record explicit. We want to be able to represent existing metadata provenance information in a simple and unified way that fits in with the DCMI context. Moreover, we want to provide provenance information for Dublin Core metadata in a DCMI compatible way.

The current *Dublin Core Abstract Model* (DCAM) is a data model for metadata applications that specifies the components and constructs used in Dublin Core metadata, independent of any particular encoding syntax (Powell, Nilsson, Naeve, Johnston, & Baker, 2007). As such, DCAM is similar to RDF and indeed it is currently not clear how DCAM relates to RDF in the future, even its deprecation in favor of RDF has been discussed (Baker & Johnston, 2010). Regarding its role, Harper (2010) states:

> Some argue that DCAM tried to be too many things to too many people. To those who understood RDF, the additional value was hard to see. Why not just use the RDF data model as the data model? To those who were not already steeped in the terminology and concepts of the Semantic Web, it

> was a dense and impenetrable document. [...] If the DCMI revises DCAM to be more closely aligned with RDF and to still apply more broadly to other encodings and syntaxes, the current document's very useful constructs will continue to add value to the metadata conversation.

The DCAM revision currently takes place in the DCMI Architecture Forum[6] and is partly driven by results from the DCMI Metadata Provenance Task Group,[7] which is co-led by the author of this thesis. Based on preliminary results, this task group has proposed a DCAM extension in line with the discussions of the RDF Working Group about the representation of g-boxes (Eckert et al., 2011). The extension is based on RDF, i.e., we assume that a mechanism to identify RDF triple sets is available and that it is therefore possible to describe these triple sets. For us, the compatibility with RDF, i.e. the implementability of DCAM within RDF is crucial. We argue that this is best achieved when DCAM is formulated in RDF.

4.3.1 DCAM vs. RDF: Abstract Model or Metadata Ontology?

This argument is contrary to the common interpretation according to which DCAM is more abstract than RDF and RDF is "only" one way to implement and serialize Dublin Core metadata, besides others like XML or plain text. Consequently, a recommendation exists how to express Dublin Core metadata in RDF (Nilsson, Powell, Johnston, & Naeve, 2008).

RDF, however, is clearly not a simple serialization, it is rather a formal language to create concrete data models that in turn can be related to each other. It is a meta model, a model to create models. This is very abstract and indeed it is questionable why a possibly even more abstract model is needed. Moreover, it is arguable, if DCAM in its current state is actually more abstract than RDF. For example, DCAM references RDF for the definition of its formal semantics and it uses URIs for the identification of resources and properties. This close relationship to RDF makes the role of DCAM unclear for people familiar with RDF.

On the other hand, there are good reasons for DCAM as a less abstract model formulated in RDF.[8] The notion of a (named) RDF graph (g-box) is very technical and

[6] http://dublincore.org/groups/architecture/

[7] http://dublincore.org/groups/provenance/

[8] This is the position of the author of this thesis, not a consensus of the DCMI Architecture Forum or the DCMI Metadata Provenance Task Group. This thesis functions as discussion paper and is therefore part of the ongoing work towards a DCAM revision.

general. The strength of RDF is that things can be identified and classified. Metadata applications deal with metadata units derived from the traditional notion of a metadata record. In DCAM, this is the *Description Set*. A description set is more specific than an RDF graph, for example, it implies that a complete set of metadata statements about a resource is contained. The definition of valid description sets for a metadata application is addressed with *Application Profiles* in Dublin Core (Coyle & Baker, 2009). Application profiles have been formalized in the *Singapore Framework* (Nilsson, Baker, & Johnston, 2008), which introduces the notion of *Description Set Profiles* (DSP). A DSP provides a template for description sets that defines mandatory elements for description sets within an application profile (Nilsson, 2008, Working Draft). Therefore, a description set should be a resource in RDF, an instance of a class for description sets, e.g., `dcprov:DescriptionSet`.[9] Analogous to other ontologies, for instance SKOS (Semantic Web Deployment Working Group, 2009) for Knowledge Organization Systems, DCAM could be an ontology for metadata. Like reification with `rdf:statement`, it would give a fundamental element in RDF a name and would allow us to talk about it in a concrete context.

4.3.2 Extending DCAM for Metadata Provenance

In the remainder of this section, we recapitulate the proposed DCAM extension, as published by the DCMI Metadata Provenance Task Group in (Eckert et al., 2011). The main objective of the group is to provide the means and guidelines to model and handle metadata provenance. The approach followed for this task has been to create a model as simple as possible, providing real world examples and mappings to other provenance approaches and comparing the complexity of the outcomes.

The work of the task group focuses on two aspects of the representation of metadata provenance. First, a domain model is needed that allows to talk about metadata. Second, a vocabulary is needed to properly describe the provenance of the metadata. The domain model forms the abstract framework that relates the provenance information to existing metadata and especially relates the classes that are introduced in the model to the existing classes in the DCAM.

[9] This class does not (yet) exist in the DCAM namespace. We use `dcprov:` as namespace prefix for proposed classes in a revised DCAM.

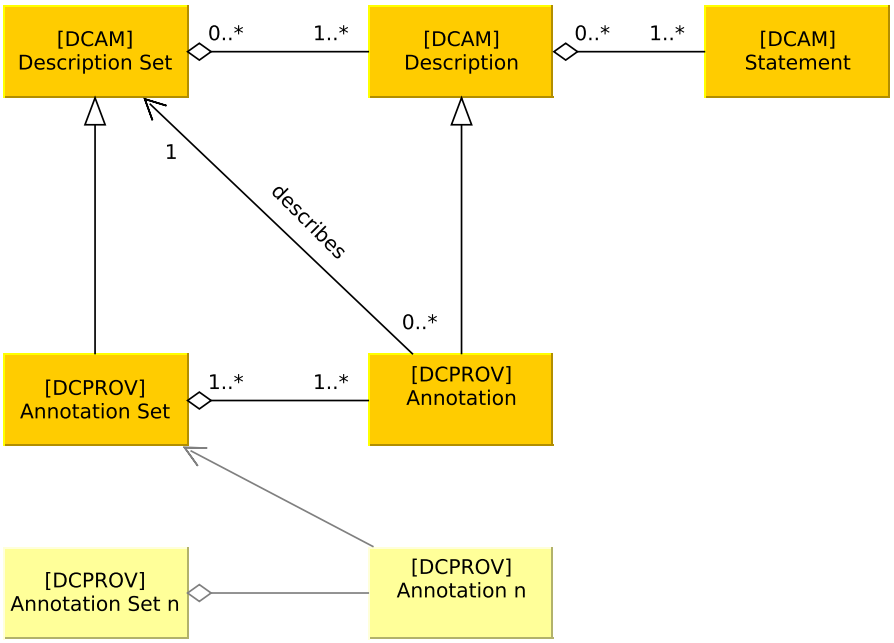

Figure 4.4: UML class diagram of the domain model

Domain model. The proposed model extends the Dublin Core Abstract Model. In particular, it uses the following classes:

- Description Set (from DCAM terminology, `dcprov:DescriptionSet`): A set of one or more Descriptions, each of which describes a single resource.
- Description (from DCAM terminology): One or more Statements about one, and only one, resource.
- Statement: A statement about a resource, according to Definition 2.3.
- Annotation: One or more Statements about one Description Set. Subclass of Description.
- Annotation Set (`dcprov:AnnotationSet`): A set of one or more Annotations. Subclass of Description Set.

Figure 4.4 illustrates the relationships between the new classes and the existing DCAM classes as a UML diagram. So what is new here and how does the proposed model relate to the exisiting DCAM and RDF?

First, the definition of a *Statement* is slightly changed, the original definition in the current DCAM is as follows: “An instantiation of a property-value pair made up of a

property URI (a URI that identifies a property) and a value surrogate." This means that a DCAM statement is only a tuple, independent of the described resource. The connection between statements and a concrete resource is made with the *Description* in DCAM. As this is fundamentally different from the notion of a statement as a triple, we propose to unify DCAM and RDF here by adopting the definition of a statement from RDF. Note that the definition of a Description does not even change in this case, so the only actual change is that it is not possible any more to talk about statements independent of a described resource.

Second, we introduce two RDF classes, i.e., we define two new types of identifiable resources: `dcprov:DescriptionSet` and `dcprov:AnntotationSet`. Description sets exist already in DCAM, however, a concrete class was never coined for them and there is no provision for their identification. Hence there is no clear way to make statements about description sets.

Together with the class for description sets, we propose the introduction of a class for *Annotation Sets* as subclass of `dcprov:DescriptionSet`. An annotation set is a description set, but it is special in the way that it talks about other description sets. This distinction is not strictly required, but we consider it convenient to indicate that way the meta-level. The definition of *Annotation* as a special description follows accordingly.

In summary, the domain model shows (1) ways in which the new entities Annotation and Annotation Set relate to and extend the existing Dublin Core Abstract Model (DCAM) entities, (2) how an annotation should be associated with the metadata it provides provenance information about, and (3) how annotations are gathered into annotation sets. The domain model is independent of the employed vocabulary that is used to create the annotations, i.e. the provenance statements.

The metadata provenance annotation. According to the domain model, annotations and annotation sets are specifications of their DCAM counterparts, i.e., subclasses in an RDF model. Just like a description set is an aggregation of descriptions (statements about a single resource), an annotation set is an aggregation of annotations (statements about a single description set) – one difference being a change in cardinality of this relationship, the motivation of which will be explained below.

This means that every annotation set is also a description set in the sense of the DCAM, and can be treated as such. If that is the case, however, why not just stick with

the DCAM entities to deal with metadata provenance instead of introducing two new key entities?

With the derivation of subclasses from DCAM we want to reflect the fact that annotations are special kinds of descriptions, because they are *only* concerned with description sets, not arbitrary resources. With this distinction of annotations and the grouping in annotation sets, we make the (provenance) annotations identifiable and also easily retrievable given a known description set.

Connecting annotations and description sets. Annotations are associated only with description sets, which in turn contain one or more descriptions. The relationship between annotations and description sets (the "role" of annotations in UML terms) is generically stated as being descriptive. The concrete mechanism or vocabulary element employed here to further specify this relationship will depend on the metadata or resource description model used in a specific metadata application or use case (e.g., RDF). The "describes" relationship in the diagram must not be confused with a specific property in RDF. In an RDF implementation, the "describes" relationship would manifest itself merely by the fact that the description set is used as a subject for the triples that form the annotations, independent of the specific relationships or properties used for these triples. Therefore, it makes sense to define that a description set – especially an annotation set – can not be empty, as the connection of an annotation set and the annotated description set is only made by the contained annotations.

The cardinality of 1 of the association on the side of the description set indicates that an annotation must only be related to a single description set. The same annotation cannot be associated with more than one description set for two reasons. On the one hand, it has to be compliant with the DCAM definition of description – "statements about one, and only one, resource"– which annotation is derived from, on the other hand, to make expressions of the domain model in metadata frameworks like RDF easier, where one annotation about two different description sets would result in two completely different triples.

Annotations are aggregated in annotation sets, just as descriptions are generally aggregated in description sets. The main difference between these can be found, once more, in cardinality. Whereas the association of a description with a description set is optional, this does not hold for the association between an annotation and an annotation

set. An annotation has to be part of at least one annotation set; conversely, every annotation set aggregates at least one annotation.

The rationale for this cardinality constraint is mainly to facilitate basic discoverability of annotations. Since (1) a variety of relationships can be used for annotating (i.e., *describing*) description sets, and (2) not all entities associated with a description set in that manner may be metadata provenance related, the annotation set as a container or wrapper has to provide a general means of retrieving metadata provenance information.

In addition, this constraint ensures that metadata provenance information can be further annotated by associating higher-level annotations with a lower-level annotation set, as seen in the lower row of Figure 4.4. Since an annotation set is a description set, it can itself be annotated by associating a further annotation set, i.e., it can capture provenance information for annotation sets as well. In this way, the model is able to handle an arbitrary number of levels.

4.3.3 A Provenance Application Profile

While the domain model outlines a mechanism that enables connecting an annotation with the annotated data, it does not describe the makeup of an annotation set for the specific context of metadata provenance, i.e., it does not provide an element vocabulary needed to put together and validate a concrete metadata provenance annotation set, but rather the generic scaffolding to accommodate such an element vocabulary.

To foster interoperability between metadata applications, DCMI uses application profiles. While the DCAM extension is a prerequiste to talk about metadata, still further information is needed what should be said about metadata to represent its provenance.

As is common practice in other application profiles, the resulting element vocabulary for creating actual annotations will most likely consist of a mix of common Dublin Core terms to state basic provenance information like creator, creation date, sources, contributors, etc., mixed with terms from other provenance vocabularies like PROV or domain-specific specializations.

The development of a provenance application profile within the Metadata Provenance Task Group will take place when the DCAM revision, PROV, and the next RDF version are finished. Preliminary work towards an application profile includes the mapping of Dublin Core terms to PROV-O (cf. Section 3.3). For the time being, the element vocabulary can be assumed to be a subset of Dublin Core, as described in Section 3.1.

4.3.4 RDF Implementation

No matter if RDF is seen as the formal language to describe DCAM or as one concrete implementation of the abstract model, the relation between DCAM and RDF has to be made explicit. As stated above, the main demand that is placed on the underlying model is the possibility to represent a description set (including regarding it as a resource in its own right). We have seen that RDF provides at least two different ways to provide statements about statements: reification and named graphs. In anticipation of the next RDF version with a clear definition of identifiable RDF graphs (g-boxes), we want to demonstrate how DCAM in RDF might look like.

Therefore, we provide an implementation based on named graphs. However, as a stopgap measure until named graphs become fully available in RDF outside of SPARQL, this could, for example, also be achieved with implicit graphs by means of the URL that is used to provide and identify the actual RDF data set, as described in Section 4.2.1.

Assume a metadata record for the *Mona Lisa*, which was – a well known fact – created by Leonardo da Vinci. But of course, Leonardo da Vinci did not create the metadata record, which in our example was created by the *Directions des Musées de France* (DMF). We use two graphs (or two RDF datasets with different URLs on the web) to define a description set and an annotation set according to our domain model (see also Figure 4.5):

```
# Named graph:  http://example.org/ML-Desc
ex:MonaLisa dct:format dctype:StillImage .
ex:MonaLisa dc:creator :LeonardoDaVinci .


# Named graph:  http://example.org/ML-Anno
ex:ML-Desc dc:creator ex:DMF .
ex:ML-Anno a dcprov:AnnotationSet .
ex:ML-Desc a dcprov:DescriptionSet .
```

The following table shows how some of the RDF resources map to their corresponding UML classes of the domain model.

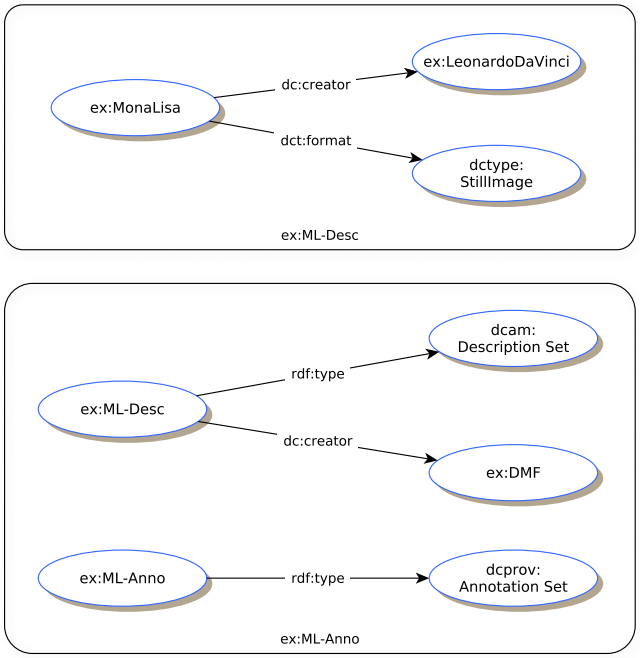

Figure 4.5: Example of an RDF implementation

RDF	UML
`ex:MonaLisa dc:creator ex:LeonardoDaVinci .`	Description
`ex:ML-Desc dc:creator ex:DMF .`	Annotation
`ex:ML-Desc`	Description Set
`ex:ML-Anno`	Annotation Set

Our example consists of two statements about the resource `ex:MonaLisa`, one about the creator of the resource, the other about its format. The graph `ex:ML-Desc` containing these statements forms a description set. Annotations about this metadata are contained in a second graph, `ex:ML-Anno`, forming an annotation set.

Statements that are part of this graph are considered annotations, i.e., statements about the provenance of the metadata of the original resource `ex:MonaLisa`, not about the resource itself. The statement "`ex:ML-Desc dc:creator ex:DMF.`" means that the Directions des Musées de France created the description of the `ex:MonaLisa` (i.e., its metadata) contained in the graph `ex:ML-Desc` as opposed to the creation of the `ex:MonaLisa` itself.

This example shows that the implementation of DCAM in RDF is simple and straigtforward using named graphs. But the representation of provenance information is only

one side of the coin. Therefore, in the next section, we show how provenance information can be retrieved using SPARQL.

4.3.5 Discoverability of metadata provenance

Given a metadata statement a, the model has to provide a path to discover if and what provanence related statement have been asserted for a. In RDF, even a known individual triple may be part of several graphs (i.e. description sets), only some of which might have been annotated. Discovery is therefore a two-stage process. First, description sets have to be determined, where the triple is part of. Then the existence of an annotation set has to be determined for each. To assert if some provenance information exists for some interpretation of a triple, the following SPARQL query can be used:

```
ASK {
     GRAPH ?ds { :MonaLisa dc:creator :LeonardoDaVinci .  }
     GRAPH ?as { ?ds ?p ?o .
          ?as rdf:type dcprov:AnnotationSet .  }
}
```

The query will return “true” if some provenance metadata is available. To then gather more information, the query can be expanded.

```
SELECT ?ds ?p ?o WHERE {
     GRAPH ?ds { :MonaLisa dc:creator :LeonardoDaVinci .  }
     GRAPH ?as { ?ds ?p ?o .
          ?as rdf:type dcprov:AnnotationSet .  }
}
```

This query finds all provenance statements about the triple. The result shows that the metadata was created by the Directions des Musées de France:

?ds	?p	?o
<http://example.org/data/ML-Desc>	dc:creator	ex:DMF

4.3.6 OAI-PMH to DC-PROV

After the theoretical presentation of the proposed DC-PROV domain model and the RDF-based examples, we want to demonstrate the possible use by means of a real world example: the translation of provenance information included in the metadata transported via OAI-PMH (Open Archives Initiative, 2008c) into the DC-PROV model. The purpose of this example is twofold: On one hand, it should help to understand the abstract classes presented in section two and show how they can be used independently of RDF. On the other hand, it hopefully supports the idea that real world data containing some metadata provenance information can be transformed into a unified data model that – albeit with some information loss – would enable true interoperability.

An OAI-PMH dataset may or may not include provenance related information. The provenance data – called origin description – contains the following elements (Open Archives Initiative, 2002):

baseURL: the baseURL of the originating repository from which the metadata record was harvested,

identifier: the unique identifier of the item in the originating repository from which the metadata record was disseminated,

datestamp: the datestamp of the metadata record disseminated by the originating repository,

metadataNamespace: the XML namespace URI of the metadata format of the record harvested from the originating repository,

originDescription: an optional originDescription block which was obtained when the metadata record was harvested. A set of nested originDescription blocks will describe provenance over a sequence of harvests,

harvestDate: the responseDate of the OAI-PMH response that resulted in the record being harvested from the originating repository, and

altered: a boolean value which must be true if the harvested record was altered before being disseminated again.

The metadata itself can be in an arbitrary format, however, the support of Dublin Core is obligatory for an OAI-PMH interface. But in this example, we don't want to

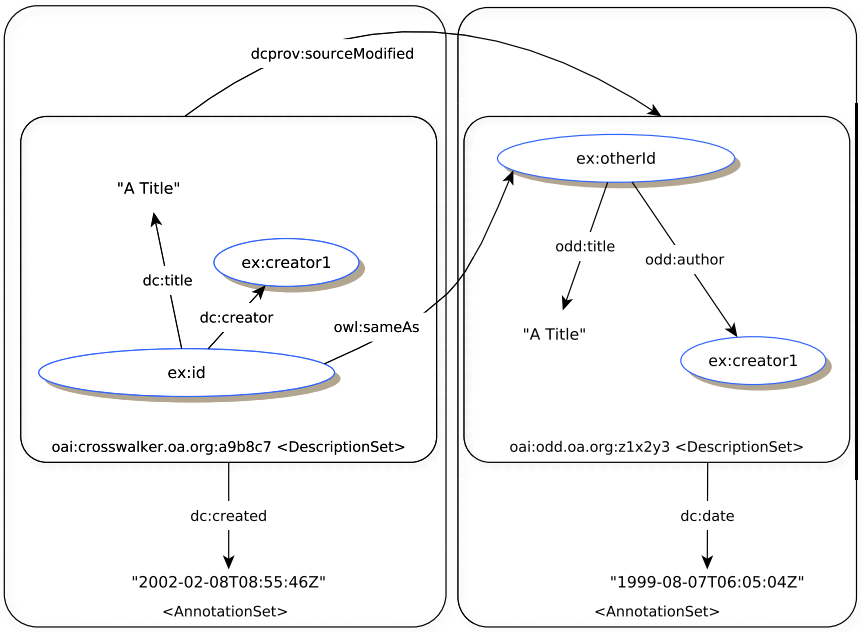

Figure 4.6: OAI-PMH translated to DC-PROV

deal with the translation of the metadata, we are concerned with the translation of the origin description.

The following example illustrates an origin description in OAI-PMH.

```
harvestDate="2002-02-08T08:55:46Z" altered="true"
baseURL = http://odd.oa.org
identifier = oai:odd.oa.org:z1x2y3
datestamp = 1999-08-07T06:05:04Z
metadataNamespace = http://odd.oa.org/odd_fmt
```

Figure 4.6 illustrates the data transformed into the DC-PROV model. As the origin description refers to a source metadata set from which the actually provided information is derived, we have to deal implicitly with two description sets, one containing the data in our PMH record, one representing the original data. The description sets are related by means of the `dcterms:source` property which is defined as "a related resource from which the described resource is derived."

In order to avoid losing the information about whether the metadata was altered since the harvesting, we propose the definition of a new subproperty of `dcterms:source`,

`dcprov:sourceModified`, which would be defined as "a related resource from which the described resource is derived by modifying it."

The identifier, according to OAI-PMH, is an identifier for the record, not the described resource. This implies that it can be used as the URI for the description set. The contents of the description sets are completely arbitrary, i.e., we are not concerned with their representation in our model. As OAI-PMH always delivers Dublin Core, it can be used straight-forwardly in this regard.

In this example, a strength of linked data becomes visible: while it is possible, it is unlikely and not practical that the whole provenance chain is transported over and over again via OAI-PMH. However, if the metadata is provided via DC-PROV, it is enough to provide one origin description with a dereferenceable URI. This way, the provenance chain remains intact and complete. As OAI-PMH is one of the most used metadata transportation formats (besides Z39.50), the comaptibility of metadata provenance representations in the Semantic Web and OAI-PMH is crucial for a wide-spread adoption in metadata applications.

4.4 Conclusion

The provenance of metadata is in theory not different from the provenance of any other resource. In this chapter, however, we have seen that the practice makes a difference. Provenance information is in itself metadata and to talk about metadata within a metadata framework can be difficult. RDF in its current version has clear limitations regarding the self-reference of its own data. On the other hand, several approaches exist and it is more than likely that the next version of RDF will provide us a standardized and clean way to talk about RDF graphs.

Even with such a mechanism at hand in RDF, however, we consider it important to have a concrete ontology that allows us to identify and classify sets of metadata statements. For this purpose, we proposed an extension of the Dublin Core Abstract Model, together with a revision that formulates DCAM in RDF. With such a metadata domain model, we would be able to (1) represent existing metadata provenance information in a simple and unified way that fits in with the DCMI context, and (2) provide provenance information for DC metadata in a DCMI compatible way.

We have demonstrated how such an RDF based DCAM could look like using named graphs and shown how our domain model can be easily adopted by content providers

in one real world example modeled with OAI-PMH. We expect it to be compliant with the notion of RDF g-boxes and PROV bundles, once PROV, the next RDF version, and the revised DCAM are finished. Regarding the specialization, the following hierarchy should hold: *g-box* > *description set* > *bundle*.

The DCAM extension is simple and minimalistic which fits to the philosophy of DCMI. It allows to represent as much (meta-)provenance levels as needed. It does so by having a simple specification following the style of DC, which is usable even if a small amount of information is lost depending on the models used in the source data.

This chapter reflects the current state of the research from the point of view of the DCMI Metadata Provenance Task Group. These results are currently incoporated in the revision of DCAM that is performed by the DCMI Architecture Forum. In the future, we aim at mapping our model to other provenance representations, be it on the structural level, like OAI-ORE, or on the description level, like PROV. We want to provide additional guidelines for publishing metadata provenance information in the form of an application profile, including potential extensions to the DC terms vocabulary for describing provenance in any domain.

5 Metadata Provenance in Europeana

Everything should be made as simple as possible, but not simpler.

Albert Einstein[1]

Europeana[2] is an internet portal that acts as an interface to millions of books, paintings, films, museum objects and archival records that have been digitized throughout Europe. Europeana aggregates digital content descriptions from cultural heritage institutions like libraries, archives, and museums. Additionally, it is a platform for knowledge exchange among professionals in the heritage sector that promotes collaboration between librarians, curators, archivists, and the creative industries. The technical accessibility and reusability of data by means of application program interfaces in Europeana plays an important role and distinguishes Europeana from other digital libraries (Concordia, Gradmann, & Siebinga, 2010). Europeana is mainly funded by the European Commission and European countries. The governing body of Europeana is the Europeana Foundation, incorporated under Dutch law as Stichting Europeana and housed within the Koninklijke Bibliotheek, the national library of the Netherlands.

In April 2012, Europeana provides access to over 23 million objects from more than 2,200 institutions in 33 countries. The content is very heterogenous, covering multiple domains. For a proper integration, all the metadata is mapped to a single data model, the *Europeana Data Model* (EDM). The EDM gradually replaces the formerly introduced Europeana's Semantic Elements (ESE) while preserving backwards compatibility.

[1] The provenance of this quote is not clear, probably it is a paraphrase of Einstein's words by Roger Sessions, cf. (O'Toole, 2011).

[2] http://europeana.eu/

5.1 The Europeana Data Model

The Europeana Data Model is described in (Europeana, 2012). A primer exists that introduces the EDM with examples and usage advices (Europeana, 2011). Doerr et al. (2010) describe the philosophy of the data model and the relationship between EDM and the old ESE.

The EDM acts as a top-level ontology, i.e., implementors in different institutions and communities are expected to create more specialized data models that are related to the EDM by means of the creation of more specific subclasses and subproperties of EDM classes and properties, respectively. A concrete mapping example – in this case from the *Encoded Archival Description*[3] (EAD) to EDM – is provided by Hennicke, Boer, Isaac, Olensky, and Wielemaker (2011).

EDM builds on RDF, i.e., all data in Europeana directly becomes available in the Semantic Web as linked open data. This is also part of the strategy to establish Europeana as a data access platform and not just as an internet portal to be used by human users. A linked open data pilot[4] has been developed, a prototypical implementation to experiment with a subset of the EDM based data that is currently available in Europeana. The prototype is described by Haslhofer and Isaac (2011).

5.1.1 Requirements

The Europeana Data Model is the basis of a framework to harvest, integrate, and expose heterogeneous metadata from thousands of cultural institutions throughout Europe. In principle, the EDM has three main requirements:

1. The EDM has to be **flexible** and **extendable** to support different specializations. A simple unified view on the metadata is needed, while the richness of domain-specific descriptions has to be preserved to make Europeana a useful tool for professionals who need to work with the data.

2. It must be possible to **relate** and **integrate** the metadata. There are various relations between the resources that are described by the metadata, in particular, often the same resources are described by several institutions. In this case, the

[3] http://www.loc.gov/ead/
[4] http://data.europeana.eu

descriptions have to be combined, as the users of Europeana are typically interested in the resources, not in a particular description.

3. It must be possible to **distinguish** the different descriptions afterwards. This is important to provide accurate information where a specific resource is described. It is planned to enrich the data in Europeana semantically, partly by automatic tools. The distinction of intellectually obtained data and automatically derived data is also important. In short: the **provenance** of all the metadata in Europeana needs to be obtained, preserved and exposed via the EDM.

5.1.2 The current EDM

The current EDM addresses all requirements listed in the last section. The flexibility and extendability is ensured by RDF and its ability to express specialized subclasses and subproperties. The classes that are defined in Europeana are very broad: the described resources belong to the class `edm:ProvidedCHO` (Provided Cultural Heritage Object), additional general classes exist to describe events, agents, places, physical things, concepts, time spans, and web resources. The classes are related to common ontologies like SKOS or CIDOC CRM. Similarly, the properties are also very general, mainly Dublin Core terms are used, with some additional specializations that are subproperties of Dublin Core terms.

The possibility to relate and integrate the metadata is also provided by RDF, as resources are identified unambiguously by URIs and relationships between resources, even the identity, can be expressed by additional statements, like `owl:sameAs`.

The third requirement, however, is the interesting one, as the provenance of metadata is not (yet) addressed directly by RDF. The EDM uses OAI-ORE (Section 4.2.5) as basis to distinguish several descriptions for a resource. This means that for each description of an `edm:ProvidedCHO`, an `ore:Proxy` resource is created that is described instead of the actual resource. The proxy and additional resources like digitized images of the resource are aggregated, i.e., related to an `ore:Aggregation`. The aggregation functions as a resource for a specific description, i.e., it can be seen as a metadata record. For each `edm:ProvidedCHO`, a special aggregation is created, the `edm:EuropeanaAggregation`. The data associated with this aggregation comes from Europeana, either created manually, or by automatic combination and enrichment of existing descriptions. The `edm:EuropeanaAggregation` represents the unified view on

an `edm:ProvidedCHO`. It is connected to a special web resource, the landing page, that presents all the available information for any cultural heritage object in Europeana.

Another specialty of the `edm:EuropeanaAggregation` is that it does not only aggregate the proxy resource and additional web resources, but also the other aggregations about the same resource.

Figure 5.1 illustrates the EDM with a concrete example, taken from the EDM primer (Europeana, 2011, p. 25). The `edm:ProvidedCHO` is the Mona Lisa painting by Leonardo da Vinci. Two records describe this resource, one from the *Directions des Musées de France*, one from the *Musée du Louvre*. Together with the descriptions, several digitized images of the Mona Lisa are provided, for instance the image on the right from the Directions des Musées de France.[5]

A web resource, showing the Mona Lisa painting.

The "actual" resources are highlighted in red (web resources) and white (provided CHOs, agents). The OAI-ORE resources that make the descriptions distinguishable are highlighted in yellow (proxies) and green (aggregations). It can be seen that the describing statements use the proxies as subject. The additional web resources and the provenance information about the data provider are assigned to the aggregations.

The example also shows the role of the special `edm:EuropeanaAggregation` that aggregates the two other aggregations together with an own proxy. Here, a semantically enriched representation of the `dc:creator` is provided, i.e., Leonardo da Vinci is unambiguously identified as an agent with a URI from the Virtual Internet Authority File (VIAF).[6] In contrast, the other aggregations simply provide (different) textual descriptions to identify Leonardo da Vinci. At last, a link to a landing page is provided for the Europeana aggregation.

5.2 Criticism

The use of OAI-ORE as a basis for metadata provenance has some disadvantages. Before we investigate possible objections, it has to be emphasized that OAI-ORE was chosen

[5] URI: http://www.culture.gouv.fr/Wave/image/joconde/0372/m503604_00-010164_p.jpg, © Réunion des musées nationaux.

[6] Leonardo, da Vinci, 1452-1519, URI: http://viaf.org/viaf/24604287

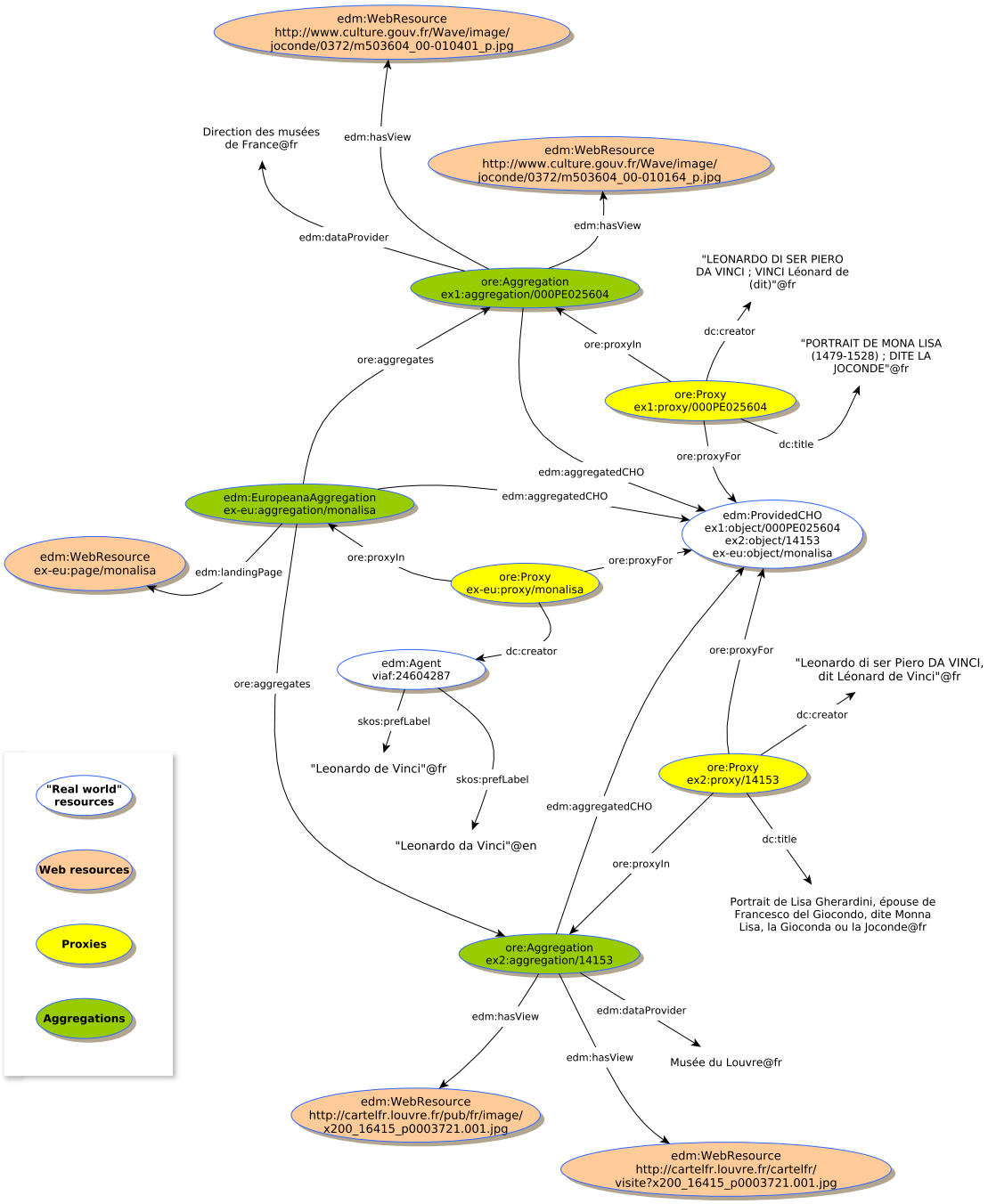

Figure 5.1: Europeana Data Model, Example

by the EDM developers due to a lack of a standardized alternative in RDF. As they state in the primer (Europeana, 2011, 6.6: Proxies vs. named graphs, p. 27):

> A question we were often asked while prototyping EDM, was why we had been considering ORE proxies to represent specific views on resources, when RDF provides the notion of "named graphs" to meet a similar requirement. The answer is quite simple, and matches the motivation for which proxies were introduced in ORE in the first place: as of the time EDM was created, named graphs were not a standard W3C recommendation, and still are not at the time this document is being written. However, the notion of graph will be present in the next version of RDF, currently being drafted by the W3C RDF Working Group. At that point, Europeana will of course consider fitting graphs into the EDM architecture.

This is indeed required as Europeana can be expected to play an important role for the Semantic Web as a huge data provider. In particular, it can be expected that the way how Europeana handles metadata provenance will affect other projects and data models as well.

So what are the problems of the current EDM? First and foremost, there are general objections against OAI-ORE as a provenance framework:

OAI-ORE provides structural means to express something within RDF that is actually not possible to express with RDF: context-dependent information about a resource that is not valid in a different context. There is nothing wrong with this approach, but it requires applications dealing with ORE data to "understand" the ORE ontology. A simple RDF application that is not aware of ORE can hardly make sense of proxies as placeholders for resources, especially it can not infer that the descriptions provided for the proxy actually refer to the original resource, albeit only in a specific context.

Another structural problem is the complex graph that originates from OAI-ORE, using additional nodes that point to other nodes in order to make complex statements. Again, there is nothing that ORE or the EDM could do about it. Without the possibility to use a metalevel, the only way to represent complex structures is by adding further nodes. Figure 5.2 illustrates this. Here, two resources are related by an arbitrary relation. A further statement should add some information about this relation, be it a provenance statement or some information about the context for which the statement holds. This can be expressed naturally with the additional metalevel, but without it, at least an additional node is needed. Furthermore, applications have to understand the meaning of

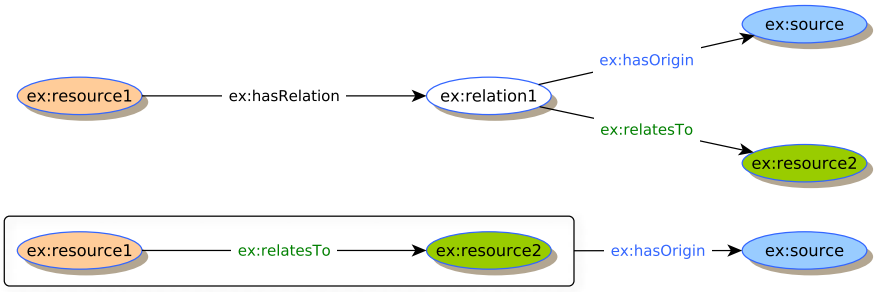

Figure 5.2: Additional nodes vs. metalevel information

the additional node, as the meaning of contextual information is not immanent. Probably worst: the actual original statement "`ex:resource1 ex:relatesTo ex:resource2.`" is lost or at least hidden in the complexer structure.

The use of such mechanisms like OAI-ORE can even lead to semantic conflicts that hinder the interoperability originally aimed for. Unfortunately, this can be shown using the example of the EDM. The following problem was identified first on the KIM-DINI-Kickoff-Workshop[7] and subsequently discussed (in German) on the LLD mailinglist of the DINI AG KIM (Eckert, 2011).

The problem arises when properties are used that have defined semantics that does not fit to the use with proxy resources. In EDM, Dublin Core terms are used to describe the provided CHOs. The semantics of Dublin Core terms is among others defined by their domain and range. For `dcterms:creator`, the following definition and range is given:

Definition: An entity primarily responsible for making the resource.

Range: `http://purl.org/dc/terms/Agent` (Definition of Agent: A resource that acts or has the power to act.)

Based on this definition, it can be inferred from a given statement "`ex:resource1 dcterms:creator ex:agent1.`" that `ex:resource1` is "the resource that has been made" and that `ex:agent1` is "the acting entity that made the resource." This also holds for the legacy element `dc:creator` that strictly speaking has to be used if not an identified resource is used as object, but a textual representation, like in our example

[7] KIM-DINI-Kickoff-Workshop, organized by DINI AG KIM and the Mannheim University Library, held at the Mannheim University Library from April 27th, 2011 to April 28th, 2011, https://wiki.d-nb.de/display/DINIAGKIM/Kick-Off-Workshop. Resolution minutes in German from the session about EDM: https://wiki.d-nb.de/display/DINIAGKIM/Session+B+-+Datenmodelle+im+Kontext+von+Europeana.

for the two statements from the museums. The definition of `dc:creator` is identical to the definition above, and despite a missing range specification, the intended meaning of `dc:creator` is made clear in the comment: "Examples of a Creator include a person, an organization, or a service. Typically, the name of a Creator should be used to indicate the entity."

With the semantics of `dc:creator` it can therefore be inferred that in general all mentions of Leonardo da Vinci and in particular the resource identified via the VIAF-URI actually refer to *the* Leonardo da Vinci. And that therefore all proxies refer to *the* Mona Lisa, as Leonardo da Vinci never created something like a proxy, but only one real Mona Lisa painting.

An application making use of the semantics by standard RDF(S) reasoning could therefore infer that all proxies are actually the same resource, which would render them useless, as a further distinction of the different descriptions would not be possible. This shows how deep OAI-ORE affects the standard mechanisms provided by RDF and the commonly applied linked data principles. Actually, only "ORE aware" applications can really make sense from ORE data. This is specifically addressed by ORE in the user guide for an HTTP implementation (Open Archives Initiative, 2008a, 6.1 Requirements for HTTP Proxy URIs):

> If an HTTP Proxy URI is used as a reference to an Aggregated Resources in the context of an Aggregation then it is desirable that dereferencing it with a standard web browser will return the Aggregated Resource itself (say a JPEG image or PDF document). In addition, dereference of the Proxy URI by an ORE aware client or agent should reveal the Aggregation context.

RDF applications would be redirected to the `edm:providedCHO`, for which no actual descriptions are available. At most there are statements that indicate that resources exist that are related to the provided CHO by `ore:proxyFor` or `edm:aggregatedCHO`.

5.3 A graph-based EDM

As the next RDF version is hopefully just around the corner, we can start to envision how the next EDM might look like. Based on the prerequisites provided in this thesis, we propose the following cornerstones:

1. Different metadata sets are technically provided as separate identifiable RDF graphs (g-boxes).
2. DCAM is used as top level ontology to describe the metadata sets, i.e., every RDF graph forms a unique, identifiable `dcprov:DescriptionSet`. DCAM is simple, does not impose any restrictions on the data and fits to the descriptive statements that are also based on Dublin Core.
3. The descriptive metadata remains unchanged, i.e., still mainly Dublin Core is used. This meets the requirement for a simple, extendable top-level ontology.
4. The provenance metadata uses Dublin Core terms as well, not PROV. PROV should be used if the creation and modification of the metadata is actually tracked, e.g., if versioning is introduced in EDM later. Until then, the easier provision of basic provenance information by means of Dublin Core reduces barriers for data providers. However, a webservice that exposes descriptive metadata and provenance metadata by means of the DC-PROV mapping would be a great add-on for the Europeana API.

Figure 5.3 illustrates how our example data would change with a graph-based EDM. We made one change in the actual descriptive data model, that certainly can be discussed: we associated the web resources directly with the provided CHO, as they are depictions of the resource, not the metadata set and therefore should be associated with the resource directly.

The definition (and the naming) of the associations colored in red would have to be changed, as they are not ORE based any more. In principle, they can be removed at all, making the data model even cleaner. The relations between the metadata sets can be derived from the graph structure, as demonstrated in Section 4.3.5. However, redundancy can improve the accessability of the data as it allows simpler and more intuitive queries.

The proposed steps have the advantage that the EDM again remains backwards compatible from the data providers point of view, with at most slight changes, if for instance the association of the web resources is changed. A cross-walk from the old model would be straight-forward, in essence all aggregations are turned in description sets and the proxies are removed, using the provided CHOs directly as subjects of all metadata statements.

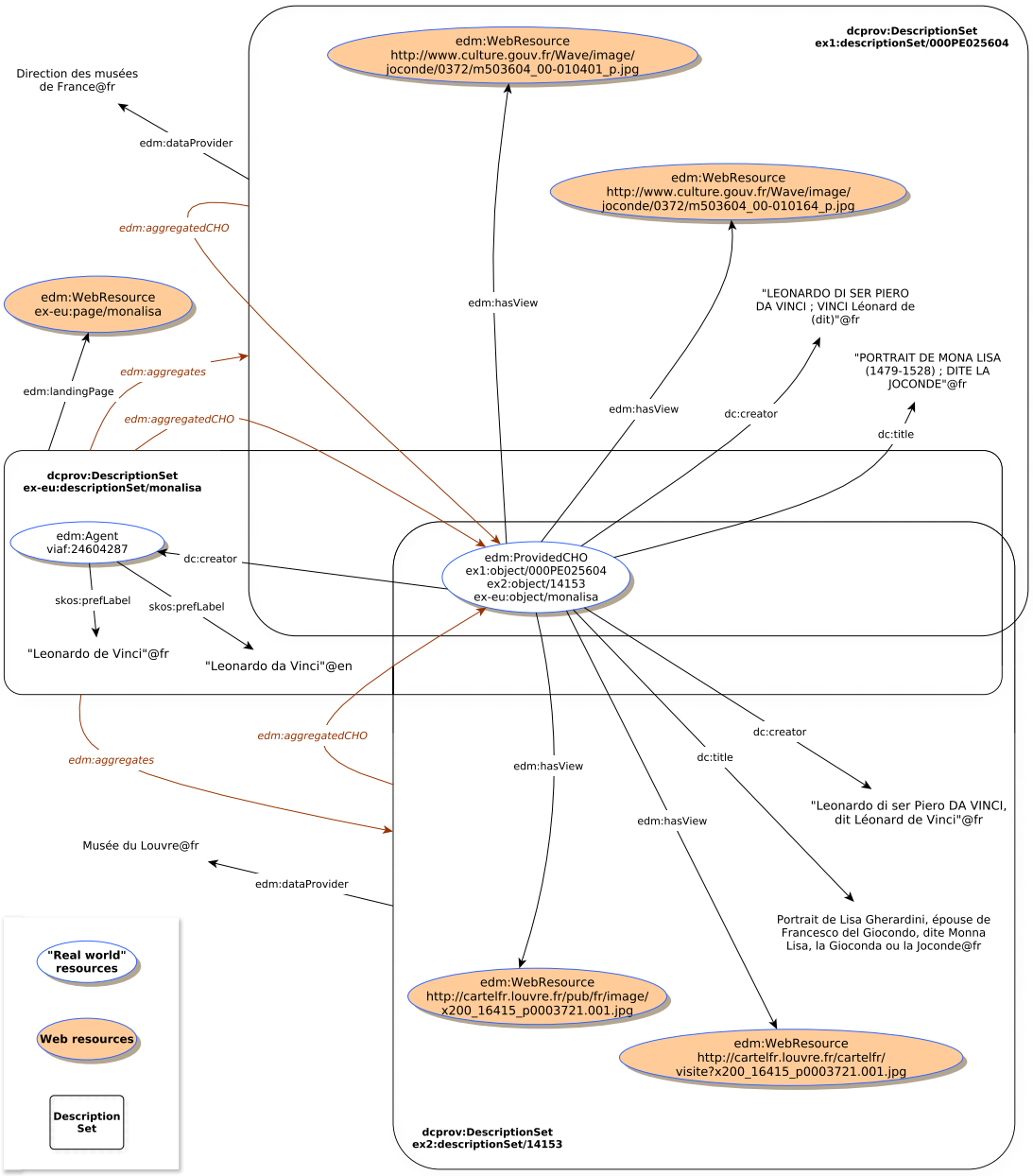

Figure 5.3: Europeana Data Model, DCAM based

6 Discussion

All problems in computer science can be solved
by another level of indirection...
except for the problem of too many levels of indirection.

David Wheeler[1]

In this thesis, we have provided an overview on metadata provenance in the Semantic Web, with a special attention to the implementation of metadata provenance in the Europeana Data Model.

We aimed at summarizing, integrating, and using different current developments that are worked on in four different working groups: the next *RDF* version supporting identifiable graph structures (W3C RDF Working Group), the *PROV* provenance data model (W3C Provenance Working Group), and an extended and revised *DCAM* abstract metadata model (DCMI Architecture Forum, DCMI Metadata Provenance Task Group). None of these groups have finished their work yet, so most of the findings in this thesis have to be taken as preliminary.

Nevertheless, it can be assumed that the outcomes of these groups more or less will be in accordance with the current state that has been used as basis for this thesis. To apply the presented approaches, in particular the proposals for a new EDM, it needs to be verified that the final outcomes are compatible. This means that RDF g-boxes, PROV bundles and DCAM description sets are compatible and can be related in form of subclass relationships (*g-box* > *description set* > *bundle*).

If this prerequisite is met, a general model for metadata provenance can be provided that we propose as basis for the next version of the EDM. In particular, we proposed the following cornerstones:

[1] cf. (Spinellis, 2007)

1. **Different metadata sets are technically provided as separate identifiable RDF graphs (g-boxes).** Not least the need for the representation of metadata provenance led to the inclusion of such a mechanism in the charter of the W3C RDF Working Group. A proper standardization will hopefully lead to a broad acceptance and to a better interoperability of applications providing and using metadata provenance information. Europeana could play a leading role here as an important implementor functioning as a role model.

2. **DCAM is used as top level ontology to describe the metadata sets, i.e., every RDF graph forms a unique, identifiable `dcprov:DescriptionSet`.** DCAM is simple, does not impose any restrictions on the data and fits to the descriptive statements that are also based on Dublin Core. DCAM could reduce the barrier for data providers and users of the Europeana API as it provides a common terminology and therefore hides the underlying technical terminology of RDF.

3. **The descriptive metadata remains unchanged, i.e., still mainly Dublin Core is used.** This is important for the backwards compatibility and also fulfills the requirements for a simple, extendable top-level ontology. This is not relevant for the topic of metadata provenance and only mentioned here for the sake of completeness.

4. **The provenance metadata uses Dublin Core terms as well, not PROV.** We recommend Dublin Core as it can be expected that the easier provision of basic provenance information reduces barriers for data providers. At the same time it raises the backwards compatibility with the current EDM.

The last point leaves the question why the new PROV ontology should not be used. Here, we have to be specific: We do not recommend PROV for the next version of EDM as Dublin Core better meets the requirements of Europeana. However, we recommend to use PROV for metadata provenance, whenever the full provenance chain of metadata has to be tracked and represented.

Another indicator for PROV would be the introduction of versioning and the relation of different versions by means of metadata provenance. Versioning is a similarly fundamental topic in the Semantic Web community as provenance and trust. Therefore, we spare a further discussion of versioning here to keep this thesis in reasonable limits.

In the introduction, we raised the following research questions that deserve a dedicated answer:

1. **How do metadata models like Dublin Core relate to more complex provenance models?** We showed that almost half of the Dublin Core terms actually provide information related to the provenance of the described resource. As a metadata vocabulary, Dublin Core focuses on the description of current facts about the resource, i.e., the relevant dates and agents that affected the resource are directly assigned. An origin of Dublin Core in bibliographic descriptions can not be neglected. Therefore, the relevant steps that led to the current state of a resource are explicitly reflected by the properties, e.g., the creation and the publication. Complex provenance models like PROV, in contrast, are concerned with the representation of the *process* that led to the current state of a resource, not the description of the resource. The most obvious difference is therefore the introduction of activities that relate an agent indirectly to a resource.

2. **Is it possible to provide a mapping between them?** In Section 3.3, we have provided a general strategy, how Dublin Core and PROV can be mapped. Therefore, the preliminary answer is yes; however, the practical applicability of such a mapping remains to be shown, when PROV and the mapping are finished.

3. **What are the general problems of metadata provenance?** In Section 4.1, we described why metadata typically is not seen as a resource of its own. The representation of metadata as an identifiable resource is a prerequisite to make statements about it and provide its provenance.

4. **How does a graph based identification of metadata records affect the representation of metadata provenance?** The graph based identification is the only straight-forward way to have identifiable metadata resources. All other approaches have to be seen as workarounds. From a modeling perspective, the representation becomes cleaner and more intuitive.

5. **Would such an approach be advantageous for the EDM?** Yes, for the reasons stated in the last answer. The understandability of the EDM is crucial for its broad acceptance.

6. **Would the use of a complex provenance model be advantageous for the EDM?** No, as described above.

We stated in the introduction that this thesis is limited to the technical representation of provenance. Nevertheless, it seems to be appropriate to conclude with some thoughts regarding the relation of legal requirements and technical feasibility for metadata provenance. One (and only one) motivation for the provision of provenance information can be that the information has to be provided due to legal obligations, e.g., because the owner of the metadata requires it. One way to solve this problem elegantly is the release of the metadata into the public domain, i.e., the owner abandons all rights. This is required by Europeana, not least to avoid interoperability problems due to legal limitations. This is indeed the best way to do it and strongly encouraged by the author of this thesis.

So do we need metadata provenance at all in Europeana, if the metadata is public domain anyway? Of course, because also in the public domain, the information is needed who created a metadata record and when and how the metadata record relates to others. Provenance is the requirement for trust and trusted data is needed to provide convincing applications on top of it.

Does the demand for the release of data in the public domain as open data become invalid, if the mechanisms for the representation of provenance are ready, as envisioned in this thesis? No, because the legal obligation to acknowledge the owner of the data would require to track the full provenance chain of all data for all times. The data could not be provided without it, it would become de facto unusable. Data is not a resource that is simply consumed, data is mixed, transformed, integrated, enriched, and improved all the time. Therefore, data has to be free.

In other words and as final conclusion: the need for metadata provenance has to be driven by the applications and data consumers, not by the data providers.

References

All online resources were checked for availability on May 25, 2012. Links to articles, where available, are given for convenience only but might no longer be available or refer to slightly different versions (pre-prints) and thus should be used with care.

Baker, T., & Johnston, P. (2010). *A review of the DCMI Abstract Model with scenarios for its future* (Tech. Rep.). Dublin Core Metadata Initiative. Available from http://wiki.dublincore.org/index.php/Review_of_DCMI_Abstract_Model (Originally written: 2010-10-15, Revised: 2011-05-12)

Beckett, D., & Berners-Lee, T. (2011). *Turtle – Terse RDF Triple Language*. W3C. Available from http://www.w3.org/TeamSubmission/turtle/

Berners-Lee, T., Fielding, R., & Masinter, L. (1998). *RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax*. IETF. Available from http://www.ietf.org/rfc/rfc2396.txt

Bizer, C., & Cyganiak, R. (2007). *The TriG Syntax*. Available from http://www.wiwiss.fu-berlin.de/suhl/bizer/TriG/Spec/

Buneman, P., Khanna, S., & Wang-Chiew, T. (2001). Why and where: A characterization of data provenance. In J. Van den Bussche & V. Vianu (Eds.), *Database theory - icdt 2001* (Vol. 1973, p. 316-330). Heidelberg: Springer. Available from http://dx.doi.org/10.1007/3-540-44503-X_20

Carroll, J. J., Bizer, C., Hayes, P., & Stickler, P. (2005). Named Graphs, Provenance and Trust. In *Proceedings of the 14th International Conference on World Wide Web (WWW) 2005, May 10-14, 2005, Chiba, Japan* (p. 613-622).

Concordia, C., Gradmann, S., & Siebinga, S. (2010). Not just another portal, not just another digital library: A portrait of Europeana as an application program interface. *IFLA Journal*, *36*, 61–69.

Coyle, K., & Baker, T. (2009). *Guidelines for Dublin Core Application Profiles*. Dublin Core Metadata Initiative. Available from http://dublincore.org/documents/profile-guidelines/

Davidson, S. B., & Freire, J. (2008). Provenance and scientific workflows: challenges and opportunities. In *Proceedings of the 2008 acm sigmod international conference on management of data* (pp. 1345–1350). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1376616.1376772

Davies, J., German, D. M., Godfrey, M. W., & Hindle, A. (2011). Software bertillonage: finding the provenance of an entity. In *Proceedings of the 8th working conference on mining software repositories* (pp. 183–192). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1985441.1985468

DCMI Usage Board. (2010a). *DCMI Metadata Terms*. Dublin Core Metadata Initiative. Available from http://dublincore.org/documents/2010/10/11/dcmi-terms/

DCMI Usage Board. (2010b). *Dublin Core Metadata Element Set, Version 1.1*. Dublin Core Metadata Initiative. Available from http://dublincore.org/documents/2010/10/11/dces/

Doerr, M., Gradmann, S., Hennicke, S., Isaac, A., Meghini, C., & Sompel, H. van de. (2010). The europeana data model (edm). In *Open access to knowledge, promoting sustainable progress – World Library and Information Congress 76th IFLA General Conference and Assembly, 10 - 15 August 2010, Gothenburg, Sweden.* Available from http://www.ifla.org/files/hq/papers/ifla76/149-doerr-en.pdf

Eckert, K. (2011). *[dini-ag-kim-lld] Europeana Data Model und ORE.* Mailinglist dini-ag-kim-lld. Available from http://lists.d-nb.de/pipermail/dini-ag-kim-lld/2011-May/000001.html (Follow-up to the discussion: http://lists.d-nb.de/pipermail/dini-ag-kim-lld/2011-July/000015.html)

Eckert, K., Garijo, D., & Panzer, M. (2011). Extending DCAM for Metadata Provenance. In *DC-2011: International Conference on Dublin Core and Metadata Applications.*

Eckert, K., Pfeffer, M., & Stuckenschmidt, H. (2009). A Unified Approach For Representing Metametadata. In *DC-2009: International Conference on Dublin Core and Metadata Applications.*

Eckert, K., Pfeffer, M., & Völker, J. (2010). Towards Interoperable Metadata Provenance. In *Proceedings of the Second International Workshop on the role of Semantic Web in Provenance Management (SWPM 2010).* Available from http://ceur-ws.org/Vol-670/ (Shanghai, China, November 7, 2010)

Europeana. (2011). *Europeana data model primer* (A. Isaac & R. Clayphan, Eds.). Europeana Professional. Available from http://pro.europeana.eu/edm-documentation

Europeana. (2012). *Definition of the europeana data model elements*. Europeana Professional. Available from http://pro.europeana.eu/edm-documentation (Version 5.2.3, 24/02/2012)

Ferris, B., & Cyganiak, R. (2011). *Reification and provenance modelling.* Mailinglist public-rdf-comments. Available from http://lists.w3.org/Archives/Public/public-rdf-comments/2011Sep/

Green, T. J., Karvounarakis, G., & Tannen, V. (2007). Provenance semirings. In *Proceedings of the twenty-sixth acm sigmod-sigact-sigart symposium on principles of database systems* (pp. 31–40). New York, NY, USA: ACM. Available from http://doi.acm.org/10.1145/1265530.1265535

Harper, C. A. (2010). Dublin core metadata initiative: Beyond the element set. *Information Standards Quarterly*, *22*, 20-28.

Haslhofer, B., & Isaac, A. (2011). data.europeana.eu – The Europeana Linked Open Data Pilot. In *DC-2011: Proceedings of the Internation Conference on Dublin Core and Metadata Applications.* Available from http://dcevents.dublincore.org/index.php/IntConf/dc-2011/paper/view/55

Hawke, S. (2011). *RDF-ISSUE-25 (Deprecate Reification): Should we deprecate (RDF 2004) reification? [Cleanup tasks].* Mailinglist public-rdf-wg@w3.org. Available from http://lists.w3.org/Archives/Public/public-rdf-wg/2011Apr/0164.html

Heath, T., & Bizer, C. (2011). *Linked Data: Evolving the Web into a Global Data Space* (1st ed.). Morgan & Claypool. Available from http://linkeddatabook.com/editions/1.0/

Hennicke, S., Boer, V. de, Isaac, A., Olensky, M., & Wielemaker, J. (2011). Conversion of EAD into EDM Linked Data. In *Proceedings of the First International Workshop on Semantic Digital Archives (SDA 2011), TPDL conference. Berlin, Germany, Sept. 29 2011.* Available from http://ceur-ws.org/Vol-801/

Hillmann, D. I., Dushay, N., & Phipps, J. (2004). Improving Metadata Quality: Augmentation and Recombination. In *DC-2004: Proceedings of the International Conference on Dublin Core and Metadata Applications.* Dublin Core Metadata Initiative. Available from http://hdl.handle.net/1813/7897

Kipling, R. (1902). *Just so stories.* Leipzig: Tauchnitz.

MacGregor, R., & Ko, I.-Y. (2003). Representing Contextualized Data using Semantic Web Tools. In *Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems, Sanibel Island, Florida.*

Moreau, L. (2010). The Foundations for Provenance on the Web. *Foundations and Trends in Web Science*, *2*(2–3), 99-241. Available from http://eprints.soton.ac.uk/271691/

Nilsson, M. (2008). *Description Set Profiles: A constraint language for Dublin Core Application Profiles.* Dublin Core Metadata Initiative. http://dublincore.org/documents/2008/03/31/dc-dsp/. Available from http://dublincore.org/documents/2008/03/31/dc-dsp/ (Working Draft)

Nilsson, M., Baker, T., & Johnston, P. (2008). *The Singapore Framework for Dublin Core Application Profiles.* Dublin Core Metadata Initiative. Available from http://dublincore.org/documents/singapore-framework/

Nilsson, M., Powell, A., Johnston, P., & Naeve, A. (2008). *Expressing Dublin Core metadata using the Resource Description Framework (RDF).* Dublin Core Metadata Initiative. http://dublincore.org/documents/2008/01/14/dc-rdf/. Available from http://dublincore.org/documents/dc-rdf/

Open Archives Initiative. (2002). *Implementation Guidelines for the Open Archives Initiative Protocol for Metadata Harvesting – XML schema to hold provenance information in the "about" part of a record* (C. Lagoze, H. van de Sompel, M. Nelson, & S. Warner, Eds.). Open Archives Initiative. Available from http://www.openarchives.org/OAI/2.0/guidelines-provenance.htm

Open Archives Initiative. (2008a). *Open Archives Initiative - Object Reuse and Exchange: ORE User Guide - HTTP implementation* (C. Lagoze, H. van de Sompel, P. Johnston, M. Nelson, R. Sanderson, & S. Warner, Eds.). Open Archives Initiative. Available from http://www.openarchives.org/ore/1.0/http

Open Archives Initiative. (2008b). *Open Archives Initiative - Object Reuse and Exchange: ORE User Guide - Primer* (C. Lagoze, H. van de Sompel, P. Johnston, M. Nelson, R. Sanderson, & S. Warner, Eds.). Open Archives Initiative. Available from http://www.openarchives.org/ore/1.0/primer

Open Archives Initiative. (2008c). *The Open Archives Initiative Protocol for Metadata Harvesting* (C. Lagoze, H. van de Sompel, M. Nelson, & S. Warner, Eds.). Open Archives Initiative. Available from http://www.openarchives.org/OAI/openarchivesprotocol.html (Protocol Version 2.0 of 2002-06-14)

O'Toole, G. (2011). Everything Should Be Made as Simple as Possible, But Not Simpler – Albert Einstein? Louis Zukofsky? Roger Sessions? William of Ockham? Anonymous? *Quote Investigator*. Available from http://quoteinvestigator.com/2011/05/13/einstein-simple/ (Blog post)

Powell, A., Nilsson, M., Naeve, A., Johnston, P., & Baker, T. (2007). *DCMI Abstract Model*. Dublin Core Metadata Initiative. Available from http://dublincore.org/documents/abstract-model

RDF Core Working Group. (2004a). *RDF Primer* (F. Manola & E. Miller, Eds.). W3C. Available from http://www.w3.org/TR/rdf-primer/

RDF Core Working Group. (2004b). *RDF Semantics* (P. Hayes, Ed.). W3C. Available from http://www.w3.org/TR/rdf-mt/

RDF Core Working Group. (2004c). *RDF Vocabulary Description Language 1.0: RDF Schema* (D. Brickley & R. Guha, Eds.). W3C. Available from http://www.w3.org/TR/rdf-schema/

RDF Core Working Group. (2004d). *RDF/XML Syntax Specification* (D. Beckett, Ed.). W3C. Available from http://www.w3.org/TR/rdf-syntax-grammar/

RDF Core Working Group. (2004e). *Resource Description Framework (RDF): Concepts and Abstract Syntax* (G. Klyne & J. J. Carroll, Eds.). W3C. Available from http://www.w3.org/TR/rdf-concepts/

RDF Data Access Working Group. (2008). *SPARQL Query Language for RDF* (E. Prud'hommeaux & A. Seaborne, Eds.). W3C. Available from http://www.w3.org/TR/rdf-sparql-query/

Runciman, B. (2006). Isn't it semantic? *ITNOW*, *48*(2), 18–21. Available from http://www.bcs.org/content/ConWebDoc/3337 (Interview with Tim Berners-Lee)

Semantic Web Deployment Working Group. (2009). *SKOS Simple Knowledge Organization System Reference* (A. Miles & S. Bechhofer, Eds.). W3C. Available from http://www.w3.org/TR/skos-reference/

Spinellis, D. (2007). Another level of indirection. In A. Oram & G. Wilson (Eds.), *Beautiful code* (pp. 279–291). Sebastopol, CA, USA: O'Reilly and Associates. Available from http://www.dmst.aueb.gr/dds/pubs/inbook/beautiful_code/html/Spi07g.html

Svensson, L. G. (2011). *Woher kommen die Daten und wie vertrauenswürdig sind sie? Provenienzmodelle für Linked Library Data. – What is the Origin of the Data and How Trustworthy are they? Provenance Models for Linked Library Data.* Unpublished master's thesis, Humboldt-Universität zu Berlin, Philosophische Fakultät I, Institut für Bibliotheks- und Informationswissenschaft.

W3C OWL Working Group. (2009). *OWL 2 Web Ontology Language Document Overview.* W3C. Available from http://www.w3.org/TR/owl2-overview/

W3C Provenance Incubator Group. (2010). *Provenance XG Final Report – W3C Incubator Group Report 08 December 2010* (Tech. Rep.). W3C. Available from http://www.w3.org/2005/Incubator/prov/XGR-prov/

W3C Provenance Working Group. (2012a). *PROV-DM: The PROV Data Model* (L. Moreau & P. Missier, Eds.). W3C. Available from http://www.w3.org/TR/2012/WD-prov-dm-20120503/ (Working Draft)

W3C Provenance Working Group. (2012b). *PROV Model Primer* (Y. Gil & S. Miles, Eds.). W3C. Available from http://www.w3.org/TR/2012/WD-prov-primer-20120503/ (Working Draft)

W3C Provenance Working Group. (2012c). *PROV-O: The PROV Ontology* (T. Lebo, S. Sahoo, & D. McGuinness, Eds.). W3C. Available from http://www.w3.org/TR/2012/WD-prov-o-20120503/ (Working Draft)

W3C SWEO Interest Group. (2008). *Cool URIs for the Semantic Web: W3C Interest Group Note 03 December 2008* (L. Sauermann & R. Cyganiak, Eds.). W3C. Available from http://www.w3.org/TR/cooluris/

Zhao, J., Bizer, C., Gil, Y., Missier, P., & Sahoo, S. (2010). Provenance Requirements for the Next Version of RDF. In *Proceedings of the W3C Workshop - RDF Next Steps, June 26-27 2010, hosted by the National Center for Biomedical Ontology (NCBO), Stanford, Palo Alto, CA, USA.*

Selbständigkeitserklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Die vorliegende elektronische Fassung ist mit der eingereichten, gedruckten Fassung identisch.

Mannheim, den 25.05.2012 Kai Eckert